Developing efficient order picker routing policies in manual picker-to-parts order picking systems

Dissertation

vorgelegt am

Fachbereich Rechts- und Wirtschaftswissenschaften
der Technischen Universität Darmstadt

von

Makusee Masae, M.Sc.

geboren in Yala, Thailand

zur Erlangung des akademischen Grades
Doctor rerum politicarum
(Dr. rer. pol.)

Referent: Prof. Dr. Christoph Glock
Koreferent: Prof. Dr. Fabio Sgarbossa

Darmstadt, 2019

Masae, Makusee: Developing efficient order picker routing policies in manual picker-to-parts order picking systems
Darmstadt, Technische Universität Darmstadt,
Jahr der Veröffentlichung der Dissertation auf TUprints: 2020
Tag der mündlichen Prüfung: 16.12.2019

Acknowledgements

The doctoral studies in Germany in the past four years are among the best experiences in my life. During the time I worked on my dissertation project, I had the assistance and the encouragement of many generous and honorable people. Among them, I would first like to express the deepest appreciation to my advisor, Prof. Dr. Christoph Glock, who gave me the opportunity to pursue my doctoral study at the Institute of Production and Supply Chain Management at Technische Universität Darmstadt. Professor Glock continually conveyed valuable support and encouragement, which made a very deep impression on me. His excellent suggestions, patient guidances, and constructive comments on my dissertation research enabled me to accomplish my doctoral studies. I have learned and gained a lot of knowledge from him both academically and personally, which will become an invaluable asset for my future academic career. I would also like to express my thanks to my external reviewer, Prof. Dr. Fabio Sgarbossa, for making the effort to read my dissertation and for writing the review report.

My doctoral studies would not have been possible without the financial support of the Department of Mathematics and Statistics, Faculty of Science, Prince of Songkla University (PSU), Thailand. I would like to express my gratitude both to the department and the university for providing me a generous financial support throughout my studies. I would also like to extend my deepest gratitude to the Institute of Production and Supply Chain Management, Technische Universität Darmstadt, for supporting me as a research assistant during the final phase of my doctoral studies.

I would like to offer my special thanks to my colleague (and my best friend) at PSU, Dr. Panupong Vichitkunakorn, for his constant help both in the academic field and my personal life. His help, comments, and suggestions on my dissertation research improved the quality of my PhD dissertation significantly.

I would further like to thank all my colleagues at the Institute of Production and Supply Chain Management for their kindness and assistance. I am very thankful to Dr. Eric Grosse for sharing his excellent background and for his constructive comments especially on systematic literature reviews. I am also very grateful to Kerstin Spiehl for her kindness that always helped me in preparing all required documents. I would like to recognize the assistance I received when I first arrived in Darmstadt from Dr. Konstantin Biel, Dr. Jörg Ries, Dr. Hamid Abedinnia, and Dr. Fabian Beck. Moreover, I am very thankful to Alexander Lunin, Nail Tahirov, Yunrong Zhang, Heiko Diefenbach, and Marc Füchtenhans for their kindness and helpful support.

Staying in Reinheim (a small town outside from Darmstadt city) is a pleasant experience and it is not a lonely life. I am particularly grateful for the friendship and the assistance given by my landlord, Walter Kießl, and my flat mates, Sahin Cangaz, Paul Sunday Wilson, and Samuel Kießl. They made my life in Germany very enjoyable. I will never forget many breakfasts, lunches, dinners, and activities we have done together.

Finally, I would like to express my deepest gratitude to my mother and my family for their unfailing support and encouragement throughout my life. My thanks also go to all my best friends in Thailand for their care, support, and encouragement during my doctoral studies in Germany.

Darmstadt, December 2019 Makusee Masae

Zusammenfassung

Die vorliegende Dissertationsschrift entwickelt effiziente Routenpolitiken für manuell betriebene Person-zur-Ware Kommissioniersysteme. Die Arbeit besteht aus insgesamt sechs Kapiteln und ist wie nachfolgend beschrieben aufgebaut. Das erste Kapitel motiviert die Themenstellung und beschreibt den Aufbau der Dissertation. Das zweite Kapitel stellt sodann die Ergebnisse eines systematischen Literaturüberblicks zu wissenschaftlichen Arbeiten zur Routenführung für Kommissionierer vor. Das Kapitel identifiziert zunächst Veröffentlichungen zu diesem Thema im Rahmen einer systematischen Literatursuche und entwickelt darauf aufbauend ein konzeptionelles Rahmenwerk, das im Anschluss für die Kategorisierung der identifizierten Forschungsarbeiten verwendet wird. Die während der Literatursuche identifizierten Arbeiten werden anschließend deskriptiv analysiert und vor dem Hintergrund des entwickelten Rahmenwerks diskutiert. Die Besprechung der Veröffentlichungen zeigt, dass eine Reihe an Forschungslücken sowohl in Bezug auf die Entwicklung exakter Lösungsverfahren als auch in Bezug auf die Weiterentwicklung von Heuristiken existiert; besonders vielversprechend erscheinen Forschungsansätze zu sein, die sich mit besonderen Eigenschaften des Kommissioniervorgangs und bislang nicht untersuchten, unkonventionellen Lagerlayouts beschäftigen. So ist ein Ergebnis des Literaturüberblicks, dass Arbeiten zur Routenführung von Kommissionierern bislang fast exklusiv angenommen haben, dass der Start- und Endpunkt der Tour identisch ist und mit dem Depot übereinstimmt; in der Praxis lassen sich jedoch Anwendungsfälle beobachten, in denen Start- und Endpunkt der Kommissioniertour andere Orte im Lager sind (etwa in Situationen, in denen die Touren während des Kommissioniervorgangs aktualisiert werden). Kapitel 3 knüpft an dieser Erkenntnis an und entwickelt ein exaktes Verfahren sowie eine Routenheuristik für ein konventionelles Lager mit zwei Blöcken, in dem der Start- und der Endpunkt der Kommissioniertour beliebige Orte sein können und damit nicht auf das Depot eingeschränkt sind. Das Kapitel erweitert damit eine frühere Arbeit von Löffler et al. (2018), die das gleiche Szenario in einem konventionellen Lager mit nur einem Block untersucht hat. Zur Lösung des Verfahrens werden die Algorithmen von Ratliff und Rosenthal (1983) sowie von Roodbergen und de Koster (2001a) adaptiert, die beide einen graphentheoretischen Ansatz sowie eine dynamische Programmierung verwenden. Das dritte Kapitel entwickelt daneben auch eine Routenheuristik, die als *S*-shape* bezeichnet wird und die im untersuchen Szenario angewendet werden kann. Die Leistungsfähigkeit beider Verfahren wird im Rahmen von Rechenstudien verglichen. Hierbei zeigt sich, dass das exakte Verfahren Touren generiert, die zwischen 6,32% und 35,34% kürzer als die heuristisch generierten Touren sind.

Eine zweite Forschungslücke, die in Kapitel 2 identifiziert wurde, bezieht sich auf das Fehlen exakter Routenverfahren für nichtkonventionelle Lagerhäuser. Das vierte Kapitel greift

eine dieser Forschungslücken auf und entwickelt ein exaktes Routenverfahren für ein nichtkonventionelles Lager, das in der Literatur als Chevron-Lager beschrieben wird. Das entwickelte exakte Verfahren baut wiederum auf den Arbeiten von Ratliff und Rosenthal (1983) sowie Roodbergen und de Koster (2001a) auf. Daneben werden drei einfache Routenheuristiken modifiziert und auf das neue Lagerlayout angepasst: Die *chevron midpoint*, die *chevron largest gap*, und die *chevron S-shape* Heuristik. Die Leistungsfähigkeit der entwickelten Verfahren wird sodann wiederum in numerischen Studien für unterschiedliche Nachfrageverteilungen und Lagerplatzvergabepolitiken untersucht. Dabei stellt sich heraus, dass das exakte Verfahren Touren generiert, die zwischen 10,29% und 39,08% kürzer als die heuristisch generierten Routen sind.

Das fünfte Kapitel beschäftigt sich im Anschluss mit einem weiteren unkonventionellen Lager, dem Leaf Lager, für das wiederum ein exaktes Routenverfahren sowie Heuristiken entwickelt werden. Auch in diesem Fall kann auf die Arbeiten von Ratliff und Rosenthal (1983) und Roodbergen und de Koster (2001a) zurückgegriffen werden, um das exakte Lösungsverfahren zu entwickeln. Die entwickelten Routenheuristiken werden als *leaf S-shape*, *leaf return*, *leaf midpoint*, und *leaf largest gap* bezeichnet. Rechenstudien, in denen die verschiedenen Verfahren für unterschiedliche Nachfrageverteilungen und Lagerplatzvergabepolitiken vergleichen, schließen das Kapitel ab. Hier zeigt sich, dass das exakte Verfahren zu Touren führt, die zwischen 3,96% und 43,68% kürzer als die heuristisch generierten Touren sind.

Das sechste Kapitel schließt die Dissertation mit einem Ausblick auf weiterführende Forschungsarbeiten ab.

Abstract

This dissertation develops several efficient order picker routing policies for manual picker-to-parts order picking systems. This work consists of six chapters and is structured as follows. Chapter 1 provides a brief introduction of the dissertation. Chapter 2 then presents the results of a systematic review of research on order picker routing. First, it identifies order picker routing policies in a systematic search of the literature and then develops a conceptual framework for categorizing the various policies. Order picker routing policies identified during the literature search are then descriptively analyzed and discussed in light of the developed framework. Our discussion of the state-of-knowledge of order picker routing shows that there is potential for future research to develop exact algorithms and heuristics for the routing of order pickers, both for order picking in specific scenarios and/or for non-conventional warehouses. One result of the literature review is that prior research on order picker routing always assumed that the picking tour starts and ends at the same location, which is usually the depot. In practice, however, it does not necessarily start and end at the same location, for example in case picking tours are updated in real time while they are being completed. Therefore, Chapter 3 proposes an exact algorithm as well as a routing heuristic for a conventional warehouse with two blocks where the starting and ending points of the picking tour are not fixed to the depot, but where they can be any locations in the warehouse instead. This chapter extends an earlier work of Löffler et al. (2018), who studied the case of a conventional warehouse with a single block, and adapts the solution procedures proposed by Ratliff and Rosenthal (1983) and Roodbergen and de Koster (2001a) that are both based on graph theory and dynamic programming procedure. Chapter 3 also develops a routing heuristic, denoted *S*-shape*, for solving the order picker routing problem in this scenario. In computational experiments, we compare the performance of the proposed routing heuristic to the exact algorithm. Our results indicate that the exact algorithm obtained tours that were between 6.32% and 35.34% shorter than those generated by the heuristic.

One of the observations of Chapter 2 is that the order picker routing problem in non-conventional warehouses has not received much attention yet. Therefore, Chapter 4 studies the problem of routing an order picker in a non-conventional warehouse that has been referred to as the chevron warehouse in the literature. We propose an optimal order picker routing policy based on the solution procedures proposed by Ratliff and Rosenthal (1983) and Roodbergen and de Koster (2001a). Moreover, we modify three simple routing heuristics, namely the *chevron midpoint*, *chevron largest gap*, and *chevron S-shape* heuristics. The average order picking tour lengths resulting from the exact algorithm and the three routing heuristics were compared to evaluate the performance of the routing heuristics under various demand distributions and storage assignment policies used in warehouses. The results indicate that the

picking tours resulting from the exact algorithm are 10.29% to 39.08% shorter than the picking tours generated by the routing heuristics. Chapter 5 then proposes an exact order picker routing algorithm for another non-conventional warehouse referred to as the leaf warehouse, and it again uses the concepts of Ratliff and Rosenthal (1983) and Roodbergen and de Koster (2001a). Moreover, it proposes four simple routing heuristics, referred to as the *leaf S-shape, leaf return, leaf midpoint*, and *leaf largest gap* heuristics. Similar to Chapter 4, we evaluate the performance of these heuristics compared to the exact algorithm for various demand distributions and storage assignment policies. Our results show that the picking tours resulting from the exact algorithm were, on average, between 3.96% to 43.68% shorter than the picking tours generated by the routing heuristics. Finally, Chapter 6 concludes the dissertation and presents an outlook on future research opportunities.

Table of Contents

List of Abbreviations

ACO	Ant colony optimization
AGV	Automated guided vehicle
ALNS	Adaptive large neighborhood search
AS/RS	Automated storage and retrieval system
BMA	Bacterial memetic algorithm
CL	Chevron largest gap
CM	Chevron midpoint
CS	Chevron S-shape
CVRP	Capacitated vehicle routing problem
EH	Ebsco Host
FW	Floyd-Warshall
GA	Genetic algorithm
HGA	Hybrid genetic algorithm
LD	Lagrangian decomposition
LKH	Lin-Kernighan-Helsgaun
LL	Leaf largest gap
LM	Leaf midpoint
LNA	Largest gap for ultra-narrow aisles and access restriction
LNAP	Largest gap for ultra-narrow aisles and access restriction plus
LR	Leaf return
LS	Leaf S-shape
MMAS	MAX-MIN ant system
MNA	Midpoint for ultra-narrow aisles and access restriction
MNAP	Midpoint for ultra-narrow aisles and access restriction plus
MST	Minimum spanning tree
O	Optimal
PC	Precedence constraints
PSO	Particle swarm optimization
PTS	Partial tour subgraph
RNA	Return for ultra-narrow aisles and access restriction
RNAP	Return for ultra-narrow aisles and access restriction plus
RR	Ratliff and Rosenthal (1983)
SA	Simulated annealing
SKUs	Stock keeping units

TS	Tabu search
TSP	Travelling salesman problem
VRP	Vehicle routing problem

List of Figures

Chapter 2

Chapter 3

Chapter 4

Chapter 5

Appendix

List of Tables

Chapter 2

Chapter 3

Chapter 4

Appendix

Chapter 1 Introduction

Supply chain management has a strong impact on a company's business success as it controls the flow of products to customers. Warehouses play a vital role in every supply chain (Tompkins et al., 2010; Öztürkoğlu et al., 2014; Richards, 2018) as they facilitate the shipping of items to the next stage of the supply chain with the highest level of customer service and at the lowest possible cost. Warehouse operations, in turn, are repetitive and labor-intensive activities, and they account for the highest share of the total logistics cost. The capital and operating costs of warehouses in the United States and Europe, for example, represent about 22-25% of the logistics costs (Baker and Canessa, 2009). It is therefore obvious that improvements in warehouse operations can contribute to the success of any supply chain.

Today's highly competitive environment and customers' wish for flexibility and quick deliveries at low cost have forced many companies to improve their warehouse operations to shorten processing times. This enables companies to satisfy customer demands quicker and to increase the overall throughput of their warehouses. The main warehouse operations that occur as part of the process of getting products into and out of the warehouse include receiving, transfer and storage, order picking, sortation, cross-docking, and shipping (de Koster et al., 2007). Among these operations, order picking is considered one of the most critical ones as it is often very labor- and time-intensive. In addition, order picking is a critical process for every supply chain because of its direct influence on customer satisfaction (Franzke et al., 2017). Underperforming in order picking can cause both unsatisfied customers and high warehouse operating costs. Consequently, improving the efficiency of order picking will lead to lower logistics costs and to an improved performance of the whole supply chain (Tompkins et al., 2010; Chen et al., 2018; Žulj et al., 2018).

In manual order picking systems, the time required by the order pickers to travel through the warehouse to reach storage locations accounts for the highest share of the total order picking time. Hence, reducing this unproductive and non-value adding time is an essential lever for improving order picking efficiency. The focus of the dissertation at hand is on the order picker routing problem in a manual order picking system. Routing the order picker optimally through the warehouse reduces travel distances and speeds up the entire order picking process. As will be shown later in this dissertation, optimal routing policies are not available for all warehouse layouts yet, which makes it difficult to efficiently operate such warehouses in practice. The dissertation aims to contribute to closing this research gap by first presenting a structured overview of order picker routing policies that have been proposed in the past, and by then developing a set of optimal order picker routing procedures for warehouse layouts for which such policies are not yet available.

Before further substantiating the research questions addressed in this dissertation, the following sub-sections provide the necessary theoretical background of order picking and order picker routing. After discussing these preliminaries, we outline the further structure of the dissertation.

1.1 Order picking

Order picking can be defined as the process of retrieving requested items from the storage locations in a warehouse in response to customer orders (Chen et al., 2015; Cheng et al., 2015; Žulj et al., 2018). The problem generally arises because incoming items are usually received and stored in large volume unit loads, while customer orders comprise small volumes of different items (Scholz and Wäscher, 2017). Therefore, the large unit loads received by the supplier have to be broken up into smaller sets, which is the basic objective of order picking.

Order picking systems that involve human order pickers can be distinguished into picker-to-parts and parts-to-picker systems. If the warehouse uses a picker-to-parts system, order pickers travel through the warehouse (afoot or using an electric cart) to retrieve items from the shelves of the warehouse. In practice, if order sizes are relatively large, order pickers may pick individual orders according to a so-called pick-by-order policy. Conversely, if the order sizes are small, several orders may be combined into batches, and the order picker would then retrieve requested items in a so-called pick-by-batch policy (Chen et al., 2018). The advantage of the pick-by-order policy is that items do not have to be sorted once the order picking process is complete, whereas pick-by-batch policies often lead to shorter average travel distances than the former policy. Given that picker-to-parts systems still rely heavily on manual human work, some researchers estimated that order picking accounts for more than 50% of the total warehouse operating costs in these systems (Frazelle, 2002; Won and Olafsson, 2005; Tompkins, et al., 2010; Bukchin et al., 2012; Pansart et al., 2018). With respect to parts-to-picker systems, the warehouse uses automated storage and retrieval systems (AS/RS) to retrieve items from storage locations and then automatically deliver those retrieved items to the order pickers at the depot. Afterwards, order pickers handle the remaining operations such as sortation, inspection, and packaging. According to de Koster et al. (2007), picker-to-parts order picking systems are still dominant in warehouses worldwide. Therefore, the dissertation at hand concentrates on analyzing picker-to-parts order picking systems.

In manual picker-to-parts systems, the order pickers are guided through the warehouse by pick-lists that specify the storage locations to be visited and the number of requested items of each product (Henn, 2012). The work elements involved in order picking may consist of travelling between storage locations, searching for requested item locations, picking items, etc.

Among them, the travelling of the order picker is usually considered the dominant component. Therefore, reducing this (mainly unproductive) travelling time is critical for any order picking system. Since the travelling distance is proportional to the travelling time, minimizing the travelling distance of a picking tour is often considered equivalent to reducing the travelling time, and it is seen as a major contributor to the improvement of order picking efficiency. The most important decision problems that have to be solved to increase order picking efficiency include *layout design*, *zoning*, *storage assignment*, *order batching*, and *order picker routing* (de Koster et al., 2007; Žulj et al., 2018; Glock et al., 2019; van Gils et al., 2019). These problems are briefly discussed in the following.

Layout design defines the size and shape of the picking area as well as the number, dimension, and the alignment of aisles (Grosse et al., 2014: Žulj et al., 2018). Examples of works that studied layout design include Caron et al. (2000), Roodbergen and Vis (2006), Roodbergen et al. (2008, 2015), Öztürkoğlu et al. (2012), Mowrey and Parikh (2014), and Öztürkoğlu and Hoser (2019). Conventional warehouses with a single or with multiple blocks are prevalent in practice and have received much attention in the literature (Henn and Schmid, 2013; Žulj et al., 2018; Chen et al., 2019). Non-conventional warehouses that have been introduced in the literature have a U-shaped (see Glock and Grosse, 2012; Henn et al., 2013), fishbone, flying-V (see Pohl et al., 2009; Gue and Meller, 2009; Çelik and Süral, 2014) or chevron layout (see Öztürkoğlu et al., 2012). Recently, also a discrete cross aisle warehouse design has been developed in the literature (see Öztürkoğlu and Hoser, 2019).

Zoning divides the picking area into zones with one order picker being responsible for each zone (de Koster et al., 2007). The literature differentiates between progressive and synchronized zoning (Yu and de Koster, 2009; de Koster et al., 2012). In the case of progressive (or sequential) zoning, orders in a batch are sequentially picked zone by zone (see Chia Jane, 2000; Yu and de Koster, 2009; Pan et al., 2015). In the case of synchronized (or parallel) zoning, order pickers in each zone can work on the same batch at the same time (see Jane and Laih, 2005; Parikh and Meller, 2008; de Koster et al., 2012).

Storage assignment determines how items should be assigned to storage locations in the warehouse (Glock and Grosse, 2012; Glock et al., 2019). The literature discusses three common strategies, namely random storage, dedicated storage, and class-based storage (de Koster et al., 2007; Gu et al., 2007; Žulj et al., 2018). For a random storage strategy, items are assigned randomly to locations available in the warehouse (see de Koster et al., 1999; Glock and Grosse, 2012; Zaerpour et al., 2013). In case of dedicated storage, items are assigned to fixed storage locations based on item characteristics such as demand frequency or volume (see Fumi et al., 2003; Glock and Grosse, 2012). In terms of class-based storage, the items are divided into classes (e.g., A, B, and C), which are then stored in dedicated areas of the warehouse. Generally, A items are the fastest moving items, while C items are slow movers. B items range in-between the two other classes. Storage assignment within each area is random (see Jarvis and McDowell, 1991; Petersen and Schmenner, 1999; Petersen et al., 2004).

Order batching refers to how customer orders should be consolidated in a single or a set of picking tours such that the (total) length of the tour(s) is minimized. The literature discusses two main batching principles, namely proximity batching and time-window batching (de Koster et al., 2007; Henn et al., 2010; Chen et al., 2018). Proximity batching combines customer orders based on their storage locations in the warehouse (items that are stored close to one another are more likely to be assigned to the same batch), whereas time-window batching consolidates customer orders that arrive during the same time interval. Examples of works that considered proximity batching include Armstrong et al. (1979), Elsayed and Stern (1983), Gibson and Sharp (1992), Gademann et al. (2001), and Henn et al. (2010). Time-window batching, in turn, was discussed by Tang and Chew (1997), Le-Duc and de Koster (2007), van Nieuwenhuyse and de Koster (2009), and Xu et al. (2014).

Order picker routing finally determines the order picker's tour through the warehouse and the sequence in which s/he retrieves requested items from the storage locations of the warehouse. A common objective of order picker routing policies is the reduction of travel distance or travel time (Elbert et al., 2017). Some researchers estimated that travel time may account for more than 50% of the total order picking time (de Koster et al., 2007; Tompkins et al., 2010). Hence, reducing this unproductive time is an essential lever for lowering warehouse operating costs.

In light of the high cost pressure many companies face in their logistics operations, and given the various technical advancements that were recently made in logistics (for example, with respect to assistive devices for order pickers such as new handhelds or augmented reality glasses), prior research has proposed various new methods for routing order pickers through the warehouse. However, as will be shown later in this dissertation[1], there are still different warehouse layouts and order picking scenarios where especially optimal order picker routing strategies have not been proposed yet. To support warehouse managers in efficiently organizing their order picking operations, the focus of this dissertation is on the order picker routing problem that will be explained in more detail in the following sub-section.

1.2 Order picker routing

Order picker routing is a variant of the classical travelling salesman problem (TSP; no capacity constraint; e.g., Ratliff and Rosenthal, 1983; Roodbergen and de Koster, 2001a; Scholz

[1] The reader is referred to Chapter 2 for a comprehensive review of the literature on order picker routing.

et al., 2016) or the capacitated vehicle routing problem (CVRP; with capacity constraint that also requires batching of orders; e.g., Lin et al., 2016; Scholz et al., 2017). Order picker routing is especially important in manual picker-to-parts order picking systems. Three general types of routing algorithms have been proposed in the literature, namely exact, heuristic, and meta-heuristic algorithms. Simple routing heuristics have enjoyed a higher popularity in the literature than optimal policies (see Section 2.6.1 for details), with frequently mentioned reasons being that they are easier to develop and to apply in practice, and that they help to avoid confusing the order pickers (Glock et al., 2017; Elbert et al., 2017). The latter aspect – the confusion of the order picker – has frequently been cited as a major drawback of the optimal policy, as the order picker may react to confusion with a deviation from the optimal route, which may lead to a longer travel time than required (Gademann and van de Velde, 2005; Glock et al., 2017). Moreover, efficient algorithms that are able to calculate an optimal route are not available for every warehouse layout and order picking scenario (see de Koster and van der Poort, 1998; Roodbergen and de Koster, 2001a; de Koster et al., 2007). This aspect also forces practitioners to use heuristics even if they should be interested in applying an optimal routing policy. Elbert et al. (2017) have shown, however, that optimal routes often lead to shorter travel distances than routes generated with the aid of heuristics even if they are subject to a higher probability of deviations from the route. Therefore, routes that are optimal from a mathematical point of view should be the preferred means of guiding the order picker through the warehouse. As mentioned earlier, order picking is responsible for more than 50% of the total warehouse operating costs, while around half of the time the order picker spends in the warehouse can be attributed to travelling. As a result, developing efficient order picker routing policies is an essential lever for lowering warehouse operating costs. Motivated by this fact, the dissertation at hand will focus on the development of new order picker routing policies.

1.3 Aim, objectives, and structure of the dissertation

The aim of this dissertation is to develop efficient order picker routing policies in manual picker-to-parts order picking systems. This aim leads to the following core research objectives of this dissertation:

1. Present a systematic literature review of order picker routing policies that have been proposed in the literature.
2. Develop an exact algorithm and a heuristic for order pickers for the two-block warehouse with arbitrary starting and ending points of a tour.
3. Develop exact algorithms and heuristics for order pickers for both the chevron and the leaf warehouses.

Objective 1 enables us to gain insights into which types of order picker routing policies have been proposed in the past, how frequently they have been applied, and for which warehouse layouts and order picking scenarios especially optimal policies are still missing. Building on the results obtained from objective 1, we will then develop optimal order picker routing policies and routing heuristics for the two-block warehouse with arbitrary starting and ending points as well as for the chevron and the leaf warehouses. The results of the dissertation at hand are valuable for warehouse managers that are interested in improving the efficiency of their order picking operations in warehouses. They are also beneficial for researchers who wish to benchmark their policies against those proposed in this dissertation, or who are interested in extending the proposed policies towards more sophisticated routing procedures or different warehouse applications.

The remainder of this dissertation is structured as follows. The next chapter presents a systematic literature review of order picker routing policies and identifies areas where further research is required. Some of the research gaps identified in the literature review define the research topics of Chapters 3 to 5. Chapter 3 considers a conventional warehouse with parallel aisles and shelves consisting of two blocks. Based on an earlier work of Löffler et al. (2018), it proposes an optimal order picker routing policy where the starting and ending points of a tour can be any location in the warehouse. Chapters 4 and 5 focus on specially structured order picking warehouses that have not yet received much attention in research on order picker routing. For both warehouse layouts, the dissertation proposes optimal order picker routing policies and alternative heuristics. The proposed routing policies are evaluated in numerical experiments for both types of warehouses. Finally, Chapter 6 concludes the dissertation and presents an outlook on future research opportunities.

Chapter 2 Review of order picker routing policies[2]

2.1 Introduction

In practice, most order picking warehouses are operated according to the picker-to-parts principle and with a high share of manual work (de Koster et al., 2007; van Gils et al., 2018), mainly because humans can more flexibly react to changes occurring in the order picking process than machines due to their cognitive and motor skills (Grosse et al., 2015; 2017). As mentioned in Chapter 1, order picker routing is one of the most important decision problems in manual picker-to-parts order picking systems. Due to its importance in warehousing and its labor-intensive tasks, researchers have proposed various routing heuristics and optimal order picker routing policies in the past that assist practitioners in optimizing order picking operations. There is a discussion in the literature about whether heuristic or optimal routing policies should be used in industry. Some researchers argued that heuristic routing policies are easier to apply in practice, and that optimal policies may confuse the order pickers, encouraging them to deviate from the optimal route (see Gademann and van de Velde (2005), Elbert et al. (2017), Glock et al. (2017) and the references cited therein). Other researchers have shown that optimal policies still perform very well even if they are subject to higher deviations than heuristic policies (Elbert et al., 2017). Aside from behavioral aspects involved in routing order pickers through the warehouse, efficient algorithms that can calculate optimal routes are not available for every warehouse layout and order picking scenario yet (see de Koster and van der Poort, 1998; Roodbergen and de Koster, 2001a; de Koster et al., 2007), which may prevent warehouse managers from improving their order picking operations in case they should be interested in doing so. An overview of order picker routing policies that supports practitioners in selecting suitable routing policies or that highlights for which order picking scenarios further ones need to be developed has, however, not been prepared so far. In addition, even though some researchers have claimed that certain routing policies are more frequently used than others, a structured investigation into the use of routing policies in the literature has never been studied so far as well. Motivated by these aspects, this chapter presents a systematic review of order picker routing policies with the following objectives:

1. Give a comprehensive overview of and characterize routing policies that have been discussed in the literature.

[2] Chapter 2 is based on the following paper: Masae, M., Glock, C.H., Grosse, E.H. (2019). Order picker routing in warehouses: A systematic literature review. *International Journal of Production Economics*, https://doi.org/10.1016/j.ijpe.2019.107564.

2. Show how frequently the routing policies have been used in the scientific literature in the past.
3. Identify seminal works that shaped the literature on order picker routing.
4. Identify warehouse layouts and order picking scenarios discussed in the literature where optimal and/or heuristic routing policies have not yet been proposed.

The intention of this review is also to stimulate further research on optimal order picker routing policies to extend the portfolio of order picker routing algorithms available to practitioners, which could in turn encourage a more extensive use of such policies in practice. Order picker routing may be connected to other order picking planning problems (e.g., order batching, zoning, storage assignment). This review does not discuss these interdependencies in detail. We argue that improving order picker routing by itself is worthwhile as more efficient order picker routing policies help leveraging the performance of integrated policies that take account of more than a single planning problem as well. For example, if we consider the joint order batching and order picker routing problem, after solving the batching problem, routes still have to be found for each batch. Hierarchical approaches for solving the joint order batching and order picker routing problem that could directly benefit from improvements in order picker routing policies are quite popular in the literature, see, e.g., Ho and Tseng (2006), Tsai et al. (2008), Chen et al. (2015) and Li et al. (2017). The reader is instead referred to the review of van Gils et al. (2018) on the interrelations of different order picking planning problems.

The remainder of this chapter is structured as follows. The next section first summarizes a seminal policy for optimally routing order pickers through a conventional warehouse. Section 2.3 then develops a conceptual framework for categorizing the literature on order picker routing policies. Section 2.4 outlines the methodology of this review and descriptively analyzes the results of the literature search. Section 2.5 presents the results of the literature review. Section 2.6 summarizes the main insights obtained in this review, and Section 2.7 concludes the chapter.

2.2 Seminal policy for optimally routing order pickers in a warehouse

A seminal work that optimally solved the order picker routing problem is the one of Ratliff and Rosenthal (1983), referred to as RR in the following, that proposed an algorithm with a time complexity that is linear in the number of aisles n (i.e., $O(n)$). Methods for efficiently solving the order picker routing problem are usually dedicated to specific warehouse layouts, and they can no longer be used in a different application. RR focused on a conventional warehouse with a single block as illustrated in **Figure 2.1**. Since the method proposed by RR has frequently been extended in the past to other warehouse layouts and/or other order picking scenarios, we briefly summarize it in the following:

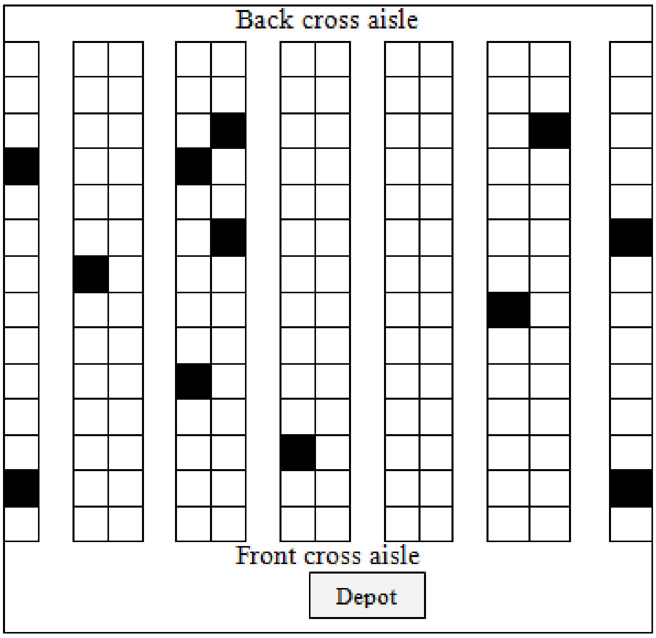


Figure 2.1 Conventional warehouse with a single block (Ratliff and Rosenthal, 1983).

The authors solved the order picker routing problem by first constructing the graph representation G of the investigated warehouse as in **Figure 2.2**. The vertex v_0 represents the location of the depot[3], where an order picker receives pick-lists and drops off completed orders. The vertices v_i for $i = 1,2, \dots, m$ correspond to the storage locations of the items requested in a customer order, whereas the vertices a_j and b_j are the rear and front ends of each aisle j, $j \in 1,2, \dots, n$. Each edge has a weight equal to the distance between the two vertices (storage locations or entry/exit points of aisles). Secondly, a sequence of increasing subgraphs $L_1^- \subseteq L_1^+ \subseteq L_2^- \subseteq L_2^+ \subseteq \cdots \subseteq L_n^- \subseteq L_n^+ = G$ is constructed. The subgraph L_j^- (and L_j^+) consists of the vertices a_j, b_j, and all vertices and edges (between and) to the left of them. A subgraph T_j of L_j^- (or L_j^+) is an L_j^- (or L_j^+) partial tour subgraph (PTS) if there exists a subgraph C_j of $G - L_j^-$ (or $G - L_j^+$) such that $T_j \cup C_j$ is a tour subgraph of G. The necessary and sufficient conditions for a subgraph T_j of L_j^- (or L_j^+) to be an L_j^- (or L_j^+) PTS are given in Theorem A.1 in the appendix. All L_j^- and L_j^+ partial tour subgraphs (PTSs) for all $j = 1,2, \dots, n$ are then iteratively constructed along the sequence using possible moves within an aisle (called vertical components V1-V6 in **Figure A.1** in the appendix) and changeovers from one aisle to the next

[3] We use the term ‘depot’ synonymously for ‘Input/Output point’ (‘I/O point’) that has also been used in the literature.

(called horizontal components H1-H5 in **Figure A.1** in the appendix). All L_j^- PTSs or L_j^+ PTSs can be grouped according to their equivalence classes, which are referred to using a triple (degree parity of a_j, degree parity of b_j, number of connected components). RR showed that any L_j^- PTS or L_j^+ PTS belongs to one of seven equivalence classes, including $(U, U, 1C)$, $(0, E, 1C)$, $(E, 0{,}1C)$, $(E, E, 1C)$, $(E, E, 2C)$, $(0{,}0{,}0C)$, and $(0{,}0{,}1C)$. The degree parities of a_j and b_j can be zero (0), even (E), or uneven (U), whereas the number of connected components can be zero $(0C)$, one $(1C)$, or two $(2C)$. In each subgraph along the sequence, if there are two or more PTSs with the same equivalence class, the one with the shortest length is kept, while the others are removed from the set of candidates for the optimal solution. The output of the dynamic programming procedure is a minimum-length L_n^+ PTS, which is a minimum-length tour subgraph of the whole graph G. RR algorithm has frequently been extended in the past, for example for conventional warehouses where a middle cross aisle separates the warehouse into two blocks (e.g., Roodbergen and de Koster, 2001a) or to the fishbone layout (e.g., Çelik and Süral, 2014). These and other extensions will be discussed in more details in Section 2.5.

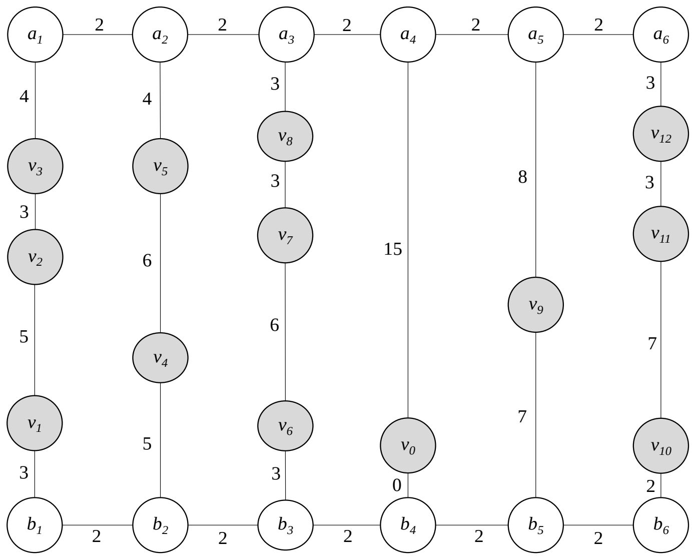


Figure 2.2 Graph representation G, where $m = 12$ and $n = 6$ (Ratliff and Rosenthal, 1983).

2.3 Conceptual framework

To characterize the order picker routing problem and the existing literature on order picker routing policies, this section proposes a conceptual framework. The framework was derived in a combined deductive and inductive approach. In the deductive approach, we developed an initial framework together with the list of keywords for the subsequent database search based on our understanding of the problem and then refined both the framework and the list of keywords inductively building on the results obtained from the preliminary review. **Figure 2.3** illustrates the developed framework. As can be seen, the framework considers two dimensions of the order picker routing problem, namely *problem characteristics* and *algorithm characteristics*. The impact of these two dimensions on order picker routing is discussed in more detail in Section 2.5. Other conceptual frameworks related to order picking were proposed by Rouwenhorst (2000), de Koster et al. (2007), Gu et al. (2007), Davarzani and Norman (2015), and Shah and Khanzode (2017), for example. These frameworks consider order picker routing as one dimension of order picking/warehousing without further analyzing its problem attributes; therefore, our work complements these earlier frameworks by going into further detail with respect to order picker routing attributes.

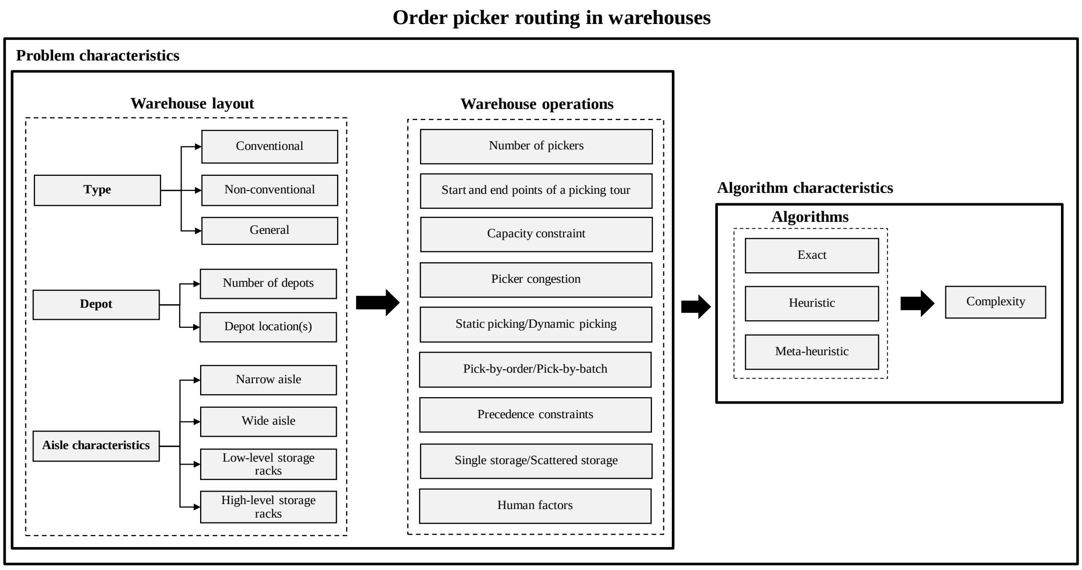


Figure 2.3 Conceptual framework used for classifying the literature on order picker routing.

2.3.1 Problem characteristics

The problem characteristics describe the order picking scenario at hand, and they include system and process attributes. They may influence the distance matrix of the order picker routing problem and may consequently impact the computational complexity of an eventual solution procedure. The framework dimension *problem characteristics* was further divided into the sub-dimensions *warehouse layout* and *warehouse operations*.

The *warehouse layout* takes account of the general type of warehouse considered, the number and location of the depot(s) and several aisle characteristics. As to the type of warehouse, the literature discussed three main warehouse variants:

- *Conventional warehouses* have a rectangular shape with parallel picking aisles that are perpendicular to a certain number of straight cross aisles. Conventional warehouses with two cross aisles on the front and back ends are often referred to as single-block warehouses (see **Figure 2.1** for an example), while warehouses with more than two cross aisles are often referred to as multi-block warehouses, where each block in the warehouse consists of a number of sub-aisles.
- *Non-conventional warehouses* do not arrange all picking aisles or cross aisles in parallel to each other, but select a different layout to facilitate reaching certain regions of the warehouse or to improve space utilization. Examples include the fishbone and the flying-V (Çelik and Süral, 2014) and the U-shaped (Glock and Grosse, 2012) layouts.
- Models of *general warehouses* do not make any assumptions about the aisles of the warehouse, but instead use general distance matrices. As a result, it is not possible to utilize specially structured distance matrices as in the work of RR, for example, which makes it difficult to solve the order picker routing problem in these warehouses efficiently. The resulting problem is identical to the classical TSP or CVRP. Examples include the works of Singh and van Oudheusden (1997) and Daniels et al. (1998).

The warehouse layout defines the number and location of the depot(s) as well as aisle characteristics. Both single- and multi-depot warehouses with wide and narrow aisles were discussed in the literature. In warehouses with narrow aisles, for example, the order picker can pick items from both sides of the aisle without having to cross it, whereas in wide-aisle warehouses, picking from both sides of the aisle makes crossing the aisle necessary leading to an additional travel distance. If the warehouse uses low-level storage racks, items can be picked directly from the racks without requiring vertical travels (Scholz and Wäscher, 2017), while in the case of high-level storage racks, vertical movements may be necessary as well. The former warehouse is usually referred to as a low-level order picking system, whereas the latter is known as a high-level system.

The sub-dimension *warehouse operations* captures various strategies employed or scenarios encountered in routing the order picker through the warehouse. It determines, for example, the number of workers picking orders in the warehouse, possible starting and ending points of a tour, and whether or not a capacity constraint has been defined for the order picker (e.g., in terms of weight or number of items; see Glock and Grosse (2012) and Matusiak et al. (2014), for example). If more than a single order picker works in the same narrow aisle, picker congestion (or picker blocking) may occur within aisles, which may induce waiting times or the need to change a picking tour while the aisle or shelf is blocked by another order picker (e.g., Franzke et al., 2017). Static order picking is an operation where pick-lists are not allowed to be changed once the picking process has been initiated, whereas in case of dynamic order picking, pick-lists may be changed during the order picking process. Our framework also considers whether the warehouse is operated according to a pick-by-order or a pick-by-batch policy. In the first case, the order picker would pick individual orders, whereas in the second case, multiple orders would be combined in a batch to reduce travel distances. In the framework, we also consider whether the pick sequence is governed by precedence constraints (PC), e.g. in case heavy items have to be picked before light items. A single storage system deals with the case where an item is stored only in a single storage location, whereas in a scattered storage system, an item is stored in multiple storage locations. Finally, our framework determines whether human factors are taken into account in the order picker routing problem. Human factors thereby describe all aspects of the design of a system (in our case: the order picking warehouse) that affect the interactions between the human and the system with the overall aim of maximizing human well-being and system performance (IEA Council 2014).

2.3.2 Algorithm characteristics

The second dimension of our framework considers the characteristics of the algorithm employed for solving the order picker routing problem as well as its time complexity. Three general types of algorithms have been proposed in the literature:

- *Exact algorithms* always find an optimal solution (i.e., shortest route) to an order picker routing problem. Examples include the algorithms of RR, de Koster and van der Poort (1998), and Roodbergen and de Koster (2001a,b).
- *Heuristics* are problem-dependent algorithms built according to its specifications, with the result in most cases not being optimal (Sörensen, 2015). Examples include the *traversal* (also known as *S-shape*), the *midpoint*, and the *largest gap* heuristics (Hall, 1993).
- *Meta-heuristics* are high-level problem-independent algorithms that provide a set of guidelines or strategies to find an approximate solution for the problem (Sörensen, 2015). Examples include *genetic algorithms* (*GA*; Tsai et al., 2008), *ant colony optimization* (*ACO*; Chen et al., 2013), *particle swarm optimization* (*PSO*; Lin et al., 2016), or *tabu search* (*TS*; Cortés et al., 2017).

2.4 The literature review

2.4.1 Related literature reviews

Table 2.1 gives an overview of existing literature reviews on warehouse operations. To highlight the contribution of our review, we summarize related reviews with respect to research focus, planning problems considered, review methodology, and overlap with the sample of our study. As can be seen in **Table 2.1**, our literature review is the only one with a clear focus on order picker routing. We intend to cover the entire literature on this topic, including the development of a comprehensive conceptual framework. The use of a systematic state-of-the-art literature search and selection strategy (see Section 2.4.2) led to a larger sample of works on order picker routing than covered in any of the existing reviews, which enables us to reach the research objectives formulated in Section 2.1 that were not addressed by earlier reviews.

2.4.2 Methodology

To get a comprehensive overview of the state-of-research of order picker routing, we conducted a systematic literature review based on the methodologies proposed by Cooper (2010) and applied, for example, in Seuring and Gold (2012) and Hochrein and Glock (2012). The literature search and selection strategy can be summarized as follows:

First, keywords that describe the subject of this review were defined to facilitate searching scholarly databases for relevant works. For this purpose, we created two lists of keywords, where list A relates to warehousing and list B to order picker routing and warehouse layout. List A included the keywords "order-picking", "order picking", "warehouse", "warehousing", and "picker", and List B included the keywords "route", "routeing", "routing", and "layout". To generate the final keyword list, each keyword from list A was combined with each keyword from list B (e.g. "order-picking" and "route", "order-picking" and "routeing", etc.). The final keyword list was then used to search the scholarly databases Ebsco Host (EH) and Scopus. Papers found in the database search were added to the working sample if they had one of the keyword combinations either in their title, abstract or list of keywords. In a second step, the papers identified during the database search were checked for relevance by first reading the paper's abstract and, if the abstract indicated that the paper may be relevant for this review, by reading the entire paper. In the third step, a snowball search was conducted in which all works that were cited in any of the sampled papers (backward search) as well as all works that cited any of the sampled papers (forward search) were checked in addition for relevance.

Table 2.1 Comparison of related literature reviews with the work at hand.

Author(s)	Research focus	Considered planning problems	Review methodology		Conceptual framework for order picker routing	Overlap (in absolute numbers and %) with our core* sample and our extended* sample
			Literature search	Literature selection strategy		
Cormier and Gunn (1992)	Warehousing	Layout design, Storage assignment, Batching, Routing	×	×	×	2 (3.7%), 0 (0%)
Van den Berg (1999)	Warehousing	Storage assignment, Batching, Routing and sequencing	×	×	×	4 (7.4%), 1 (0.7%)
Rouwenhorst et al. (2000)	Warehousing	Layout design, Storage assignment, Batching, Routing and sequencing	×	×	×	3(5.6%), 2(1.3%)
De Koster et al. (2007)	Order picking	Layout design, Storage assignment, Zoning, Batching, Routing	×	×	×	10 (18.5%), 17 (11.4%)
Gu et al. (2007)	Warehousing	Storage assignment, Zoning Batching, Routing and sequencing, Sorting	×	×	×	10 (18.5%), 13 (8.7%)
Davarzani and Norrman (2015)	Warehousing	Layout design, Storage assignment, Batching, Routing	✓	✓	×	2 (3.7%), 5 (3.4%)
Shah and Khanzode (2017)	Warehousing	Zoning, Wave picking, Batching, Routing, Picking equipment, Sorting, Layout and slotting, Replenishment, Picking productivity and e-fulfilment,	✓	×	×	7 (13.0%), 22 (14.8%)

Author(s)	Research focus	Considered planning problems	Review methodology		Conceptual framework for order picker routing	Overlap (in absolute numbers and %) with our core* sample and our extended* sample
			Literature search	Literature selection strategy		
Van Gils et al. (2018)	Order picking	Combination of planning problems (e.g., batching and routing)	✓	✓	×	13 (24.1%), 38 (25.5%)
Boysen et al. (2019)	Warehousing	Mixed-shelves storage, Batching, Zoning, Sorting, Dynamic order processing, AGV-assisted picking, Shelf-moving robots, Advanced picking workstations	✓	✓	×	14 (25.9%), 35 (23.5%)
This paper	**Order picking**	**Routing**	✓	✓	✓	

✓= employed in the literature review, ×= not mentioned in the literature review, * = There are 54 papers in the core and 149 papers in the extended sample of our review (see Section 2.4.2)

During the evaluation of the database search results, the following inclusion and exclusion criteria were applied to the working sample:

- Only works that study order picker routing policies for manual picker-to-parts warehouse operations were considered relevant. Works that propose travel time or average tour length estimation models for order picking, without actually proposing a routing policy, were excluded from further analysis. Examples are the works of Parikh and Meller (2010), Mowrey and Parikh (2014), and Venkitasubramony and Adil (2016). Similarly, also works that investigate the routing of AS/RS, automated guided vehicles (AGVs), robots or tow trains through a warehouse were excluded from further analysis (e.g., Gademann, 1999; Fazlollahtabar et al., 2015; Boysen et al., 2017).
- We differentiated between a 'core sample' and an 'extended sample'. For the core sample, only works that either proposed a particular order picker routing policy or combined two existing routing policies contained in the core sample in a hybrid method for the first time were considered relevant. The extended sample, in turn, considers all works that apply routing policies proposed in the core sample. Works that both proposed a new routing policy and used an existing routing policy already contained in the core sample were assigned to both samples. For example, the work of de Koster and van der Poort (1998) developed both an optimal order picker routing policy and used the *S-shape* heuristic proposed by Hall (1993). This paper was consequently assigned to both the core and the extended samples. The core sample is discussed in detail below, which enables us to give a comprehensive overview of all order picker routing policies that have been proposed in the literature so far. An additional analysis of the extended sample then enables us to derive insights into the frequency and context of usage of the different routing policies in the academic literature.
- Only works that appeared in peer-reviewed journals were considered relevant. Thus, so-called grey literature such as book chapters, conference papers, theses, technical reports, etc., were excluded from the review.
- Only works written in English were considered relevant.

The results of the literature search are illustrated in a review protocol in **Table 2.2** (all numbers effective June 2019). The database search resulted in 337 papers from EH and 735 papers from Scopus. An analysis of the papers' abstracts reduced the size of the core samples to 62 (EH) and 62 (Scopus) papers, respectively. After eliminating duplicate papers, 73 papers remained in the core sample. Reading all papers led to the exclusion of 37 papers and a core sample consisting of 36 papers. Backward and forward snowball searches helped to identify 10 additional papers. Finally, discussions with experts helped to identify another 8 papers that had been missing in the core sample, which led to a final core sample size of 54 papers. With respect to the extended sample, a total number of 149 papers were identified, 44 papers from EH, 34 from Scopus, and 71 from the snowball search. Note that papers included in the extended sample that were not cited in the text are listed in the appendix.

Table 2.2 Review protocol for the core sample.

Filter type	***Descriptions and guidelines***	***Results***	
Inclusion criteria	*Topic:* Papers with a focus on order picker routing in a warehouse *Peer-reviewed journals:* Academic journal papers *Language*: Limited to English *Time span*: Not limited		
Keywords combination	"order picking and route" or "order picking and routeing" or "order picking and routing" or "order picking and layout" or "order-picking and route" or "order-picking and routeing" or "order-picking and routing" or "order-picking and layout" or "warehouse and route" or "warehouse and routeing" or "warehouse and routing" or "warehouse and layout" or "warehousing and route" or "warehousing and routeing" or "warehousing and routing" or "warehousing and layout" or "picker and route" or "picker and routeing" or "picker and routing" or "picker and layout"		
Keyword search	Search selected online databases with the keyword combinations defined above.	***EH***	***Scopus***
	Ensure substantive relevance by requiring that all papers contain at least one keyword combination in their title, abstract or list of keywords.	337	735
Consolidation I	Ensure relevance of content by subjecting all papers to a manual analysis of their abstracts.	62	62
	Results from selected databases were consolidated and duplicate papers were eliminated.	73	
Consolidation II	Ensure relevance by completely reading all papers left in the sample.	36	

Filter type	*Descriptions and guidelines*	*Results*
Snowball approach	Search for additional papers in backward/forward snowball searches based on all previously selected papers.	10
Expert consultation	Discuss search results with experts to identify missing papers.	8
	Final core sample size	**54**

2.4.3 Descriptive results

Figure 2.4 presents the number of papers published per year in the core and the extended samples. As can be seen, publication numbers of both samples displayed an increasing trend in recent years, with more than 50% of the core sample papers having been published during the last five years. This trend may point towards an increasing relevance of alternative order picker routing policies in practice, which may reflect the high cost pressure many warehouses face in industry. Furthermore, an increasing number of papers in the extended sample indicates that also the application and eventual validation of existing order picker routing policies has enjoyed popularity in recent years. **Figure 2.5** shows the academic journals that published at least two papers contained either in the core or the extended sample. The most popular outlets with at least two papers in both samples are the *International Journal of Production Research* (11 core, 23 extended), the *European Journal of Operational Research* (12 core, 19 extended), *IIE (or IISE) Transactions* (4 core, 15 extended), *Computers & Industrial Engineering* (3 core, 10 extended), and *Computers & Operations Research* (3 core, 5 extended), the *International Journal of Production Economics* (2 core, 7 extended), and *Applied Soft Computing* (2 core, 2 extended).

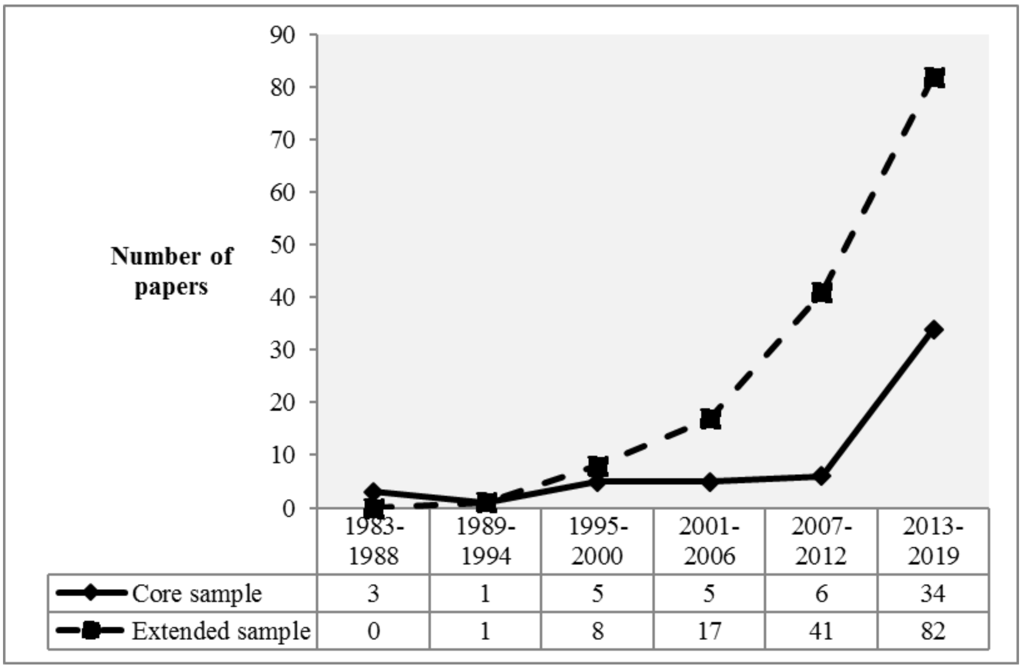


Figure 2.4 Number of core and extended sample papers published per year[4].

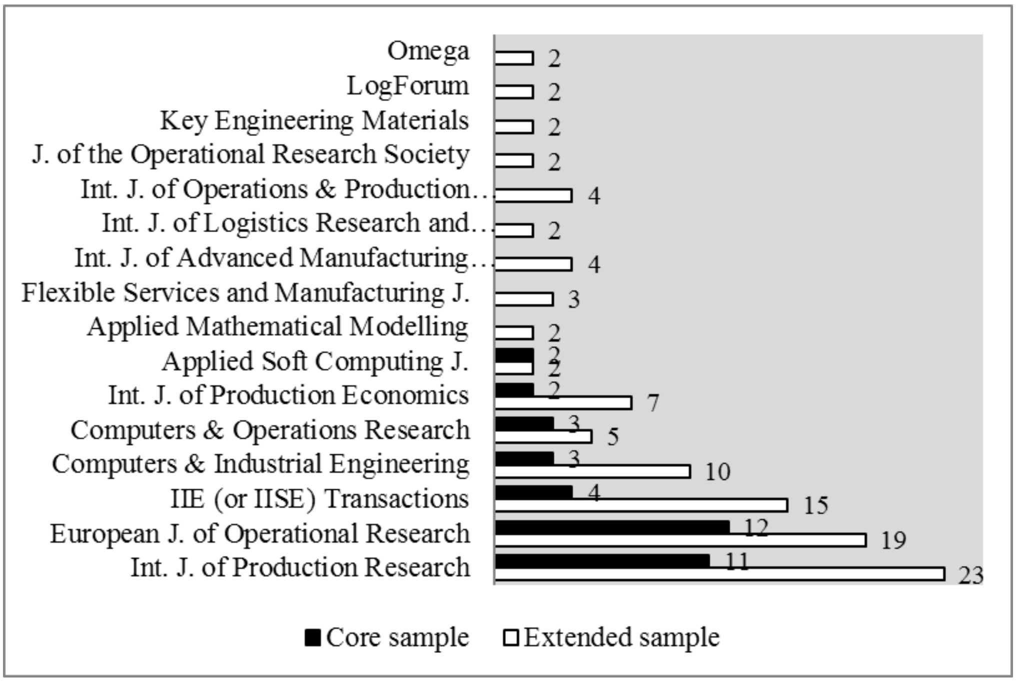


Figure 2.5 Number of core and extended sample papers published per journal.

[4] The year 2019 is only considered until and including June.

2.5 Results of the literature review on order picker routing algorithms

This section reviews algorithms for the order picker routing problem. The structure of the discussion follows the conceptual framework and categorizes algorithms according to the type of warehouse considered and the type of algorithm (see Section 2.3). The remaining components of our framework are addressed for each paper where applicable and summarized in the appendix.

2.5.1 Conventional warehouse

2.5.1.1 Single-block warehouses

This section reviews algorithms employed for solving the order picker routing problem in single-block warehouses. We structure the discussion along the algorithm types defined above.

2.5.1.1.1 Exact algorithm

A seminal work for the optimal routing of order pickers through a warehouse is the one of RR. The warehouse investigated in this work has narrow picking aisles with a single depot in the front cross aisle. The authors further assumed a low-level storage rack for static picking in a single storage system. A picking tour starts and ends at the depot, and requested items are picked according to the pick-by-order principle by a single picker. The device's capacity is sufficient for picking all requested items in a single picking tour. The time complexity of the algorithm that was already described in more details in Section 2.2 is linear in the number of aisles (i.e., $O(n)$, where n is the number of aisles). The algorithm of RR lacks flexibility to be used if order picking scenarios change, and it has therefore frequently been extended in the past. De Koster and van der Poort (1998), for example, generalized the algorithm to the case of decentralized depositing, which describes a situation where the order picker can deposit the retrieved items at the respective front ends of each picking aisle without returning to the depot, and it can be found in practice in situations where conveyor belts are used to transport picked items to the central depot or the shipping area. Thus, once an order has been completed, the order picker can proceed with the next order without having to return to the depot. As a result, the starting and ending points of a picking tour are not necessarily the depot, but instead they can be any of the front ends of the picking aisles. Permitting more than a single starting and ending point for a tour leads to new equivalence classes for the PTSs in addition to the equivalence classes proposed by RR. The algorithm's time complexity is linear in the number of aisles or items ($O(\max(n, m))$).

Another extension of RR that takes account of PC was proposed by Žulj et al. (2018). Their investigated warehouse is a single-block warehouse with a single depot located at the front of the left-most picking aisle. Note that this warehouse layout is shown in **Figure A.2** in the appendix as *standard layout A*. PC, in this context, defines partial sequences for the picking of items based on the weight, fragility, and/or item category. For each picking tour, an order picker initially retrieves all heavy items contained on a pick-list and ends the tour at a predetermined heavy item location. S/he then further retrieves all light items contained on the same pick-list and finally returns to the depot. The optimal order picker route is determined by finding a combination of heavy and light subtours that results in a total tour with minimum length. The time complexity of the proposed algorithm is $O(m^5)$.

Çelik and Süral (2016) extended the algorithm of RR by considering turn penalties in addition to the order picker's regular travel time. Assuming that changes in the direction of travel slow down the order pickers, turn penalties are encountered whenever the order picker enters or leaves an aisle or when a U-turn is necessary within an aisle. The authors considered different depot locations in their study, namely (i) at the corner of a pick aisle and a cross aisle (so-called *corner-depot*); (ii) at a cross aisle, but not at the corner of a pick aisle and a cross aisle (*cross-depot*); and (iii) at a pick aisle (*pick-depot*). The authors solved different variants of the problem (single-objective turn minimization and time minimization, bi-objective travel time and turn minimization, and a tri-objective problem with U-turn minimization) in polynomial time.

All works mentioned above only consider static order picking. Lu et al. (2016), in contrast, extended RR's algorithm to account for situations where a pick-list that is currently being completed can be updated, e.g. because new orders have arrived at the warehouse. This situation is also known as dynamic order picking. Once the pick-list has been updated, a new picking tour is calculated with the starting point of the tour being the current position of the order picker. The ending point of each tour would still be the depot. Since any arbitrary position in the warehouse could be the starting point of a new tour, edges corresponding to possible moves to leave an aisle have to be considered in RR's algorithm. This leads to new equivalence classes of PTSs in addition to the PTSs proposed in RR that have to be considered during the construction of the order picking tour. The time complexity of the proposed algorithm is $O(n)$.

Besides exact algorithms based on RR, Chabot et al. (2017) used an exact algorithm for the vehicle routing problem (VRP), namely *branch-and-cut*, to solve the order picker routing problem with PC in a wide-aisle warehouse. The authors proposed mathematical formulations derived from single- and two-index VRP models, namely (i) the *capacity-indexed formulation* (Picard and Queyranne, 1978) and (ii) the *two-indexed flow formulation* (Laporte, 1986; Toth and Vigo, 2014). A *branch-and-cut* algorithm was applied to solve the two formulations where weight and fragility inequality constraints were used as cutting planes at every node of the *branch-and-bound* tree for strengthening the linear programming relaxation. The warehouse investigated is a single-block warehouse with a single depot located half-way between the left- and right-most picking aisles (see *standard layout B* in **Figure A.2** in the appendix).

Several other exact algorithms have been developed for the case where all items requested in an order are stored in a single picking aisle, including the works of Goetschalckx and Ratliff (1988a,b) and Charkhgard and Savelsbergh (2015). Goetschalckx and Ratliff (1988a) proposed two exact routing algorithms for wide-aisle warehouses with low-level storage racks they termed (i) *optimum aisle traversal* and (ii) *optimum return*. Both algorithms consider the case where all items requested in an order are stored in a single picking aisle. For algorithm (i), the authors assumed that a picking tour starts at the entry point of an aisle and ends at the exit point at the opposite end of the aisle. The *no-skip* property that is based on the *no-crossing* property of Barachet (1957) was applied to determine an optimal order picker route for this case. The main idea of the *no-crossing* property is that a Hamiltonian path visits each vertex exactly once. As a result, such a path cannot contain a vertex with a degree other than two, hence paths will not cross themselves. The order picker always starts at the aisle entrance and then picks the nearest requested item either on the right or left side of the aisle. After that, s/he either picks the next item on the same current side of the aisle or crosses to the other side of the aisle to pick the item. The problem of finding the optimal order picker route in this case is equivalent to finding the shortest path in an acyclic graph, i.e. a graph without cycles that allows that each vertex is visited at most once. In Goetschalckx and Ratliff (1988a), each vertex represents the state of the system defined by a triple (last item picked on the right side, last item picked on the left side, current position of the order picker (right or left)), and each edge represents a feasible transition with a certain travel distance. The authors used a dynamic programming approach to find the shortest path in the acyclic graph where the travel distances for all transitions from the entry to the exit points were computed. The time complexity of the algorithm is $O(m^2)$. Charkhgard and Savelsbergh (2015) further studied algorithm (i) and calculated a minimum spanning tree (MST) on the pick locations on both sides of the aisle and then connected the entry and exit points to their closest pick locations. Using the MST in the *optimum aisle traversal* strategy, a lower bound on the length of the picking route can be computed in linear time ($O(m)$). The authors termed this routing policy the *passing strategy*. In case of algorithm (ii) discussed in Goetschalckx and Ratliff (1988a), the order picker starts at the entry of the aisle, picks all items on one side of the aisle, then crosses to the farthest item on the other side of the aisle, and picks all remaining items on the way back. Goetschalckx and Ratliff (1988b) determined the optimal number of stops of a picking device and the pick sequence at each stop in a wide aisle of a single-block warehouse. The authors formulated this problem as a set covering problem with the consecutive-ones property (see Segal, 1974 and Bartholdi and Ratliff, 1978), which can be solved by finding the shortest path in an acyclic graph.

2.5.1.1.2 Heuristics

The algorithms discussed in the previous section always find the shortest possible tour for the order picker routing problem. In many practical applications, the use of heuristics is common, which – despite their performance disadvantages – are easy to apply and which produce results that can easily be understood and implemented by the order picker. Routing heuristics for single-block warehouses can be classified as *simple heuristics* or *TSP heuristics*. The first type was specifically developed for order picking problems, whereas the second type was originally developed for the TSP and transferred to an order picking context. The *simple heuristics* discussed in the sampled papers are summarized in the following:

Goetschalckx and Ratliff (1988a) proposed a *simple heuristic* for wide-aisle warehouses. They referred to as the *Z-pick* heuristic. The heuristic determines a route where the order picker travels in a zigzag pattern through the wide aisle to collect requested items from both sides of the aisle.

Hall (1993) proposed and compared three *simple heuristics* for the order picker routing problem in a single-block warehouse with narrow aisles and a single depot, referred to as the *traversal*, the *midpoint*, and the *largest gap* heuristic. Petersen (1997) added the *return* and the *composite* heuristic. These heuristics are simple 'rules of thumb' and can be summarized as follows:

- *Traversal* (also known as *S-shape*): The order picker starts in the first aisle that contains requested items and traverses the aisle completely. The order picker then moves to the next aisle that contains requested items, traverses this aisle completely, and continues in this fashion until all requested items have been retrieved. Note that this heuristic had earlier been discussed by Kunder and Gudehus (1975).
- *Midpoint*: The warehouse is divided into two equal halves, referred to as the front and the back parts. The order picker enters the aisles in the front part of the warehouse that contain requested items, and leaves each aisle on the side where s/he entered it without accessing the back part. Once the front part of the warehouse has been completed, the order picker moves to the back part of the warehouse to complete all aisles in the same fashion.
- *Largest gap*: This heuristic also divides aisles into two halves, but uses the largest gap between two requested items or between the aisle exits and a requested item for defining the front and back part of each aisle. As in the case of the *midpoint* strategy, the order picker first completes the front part of the warehouse and then moves to the back part to collect requested items there.

- *Return:* The order picker enters each aisle that contains at least one requested item from the front end and picks all requested items. Once the order picker retrieves the last item, s/he returns to the front end of the aisle and continues to the next aisle.
- *Composite:* This strategy combines the *return* and the *S-shape* heuristics such that the order picker can either entirely traverse the aisle or return to the front end where s/he entered it, depending on which heuristic gives the shortest travel distance for retrieving the farthest requested items from two adjacent aisles.

Chabot et al. (2017) modified the heuristics proposed by Hall (1993) as well as the *combined* policy proposed by Roodbergen and de Koster (2001b) (to be discussed in detail in Section 2.5.1.2.2 as it was originally proposed for multi-block warehouses) to solve the order picker routing problem with PC. Each modified heuristic follows the original procedure with the additional condition that a requested item is retrieved only if it respects all constraints of the problem. Otherwise, it is skipped and picked in the next picking tour. Once the transport capacity of the order picker has been reached or the last item has been picked, the order picker returns to the depot to drop off all retrieved items. If further items need to be picked, the order picker starts a new tour at the first skipped item or the first unpicked item in the regular sequence. This procedure is repeated until all remaining items have been picked.

Menéndez et al. (2017) proposed another extension of the *combined* heuristic of Roodbergen and de Koster (2001b) for *standard layout A*. The proposed heuristic starts by first evaluating for each individual aisle (excluding the left- and right-most aisles) if using the *largest gap* or the *S-shape* heuristic leads to a shorter travel distance for this aisle. The heuristic then evaluates different options for combining the resulting individually shortest travel distances within an aisle, taking account of the direction from which an aisle should be entered.

With respect to *TSP heuristics*, Makris and Giakoumakis (2003) applied a modified *k-interchange* heuristic to improve the solution of a simple routing heuristic (e.g., *S-shape*). The *k-interchange* heuristic, originally proposed by Nemhauser and Wolsey (1988), is a local search heuristic that improves solutions obtained for the TSP. Given an initial tour, the *k-interchange* heuristic replaces k edges in that tour by k edges that are not in the tour if such a change yields a shorter tour. The modified *k-interchange* heuristic changes the position of two random requested items in a tour, which leads to four edges in the tour being replaced by four new edges. A repetitive application of this heuristic may reduce the length of the initial tour. Grosse et al. (2014) studied order picker routing in *standard layout B* with narrow aisles and used, among others, the *savings algorithm* for routing order pickers. The *savings algorithm*, originally proposed by Clarke and Wright (1964), starts with a set of tours in which each item is picked individually. It then evaluates the travel distance that can be saved when merging two existing tours into a single tour, and combines those tours that result in the highest saving in travel distance.

All previously mentioned heuristics are confined to order picking in a single storage system. Only few works considered order picker routing in a scattered storage system. In this scenario, Weidinger (2018) considered order picker routing in *standard layout B*. Two optimization sub-problems have to be solved in this case: (i) determine which locations to visit; and (ii) route the order picker for the set of locations determined in (i). The author proposed three routing heuristics using storage location selection rules that calculate priority values for the requested items, which influence the picking sequence of requested items in an order. Weidinger et al. (2019) extended this to the case where depots are located both at the rear and front ends of each aisle, so that the starting and ending points of a picking tour can be any of the rear and front ends of the picking aisles. They formulated a mixed-integer optimization model along with a *pool-based construction heuristic* to solve it.

2.5.1.1.3 Meta-heuristics

Meta-heuristics have mostly been used to solve combinations of multiple order picking planning problems and complex order picking problems. Works that used meta-heuristics for solving combined planning problems are discussed in the following. Tsai et al. (2008) proposed two *GAs* to solve the order batching and order picker routing problems considering both travel cost as well as earliness and tardiness penalties. The authors first constructed batches using a *GA*, and then applied another *GA* to find a short route for the order picker given a set of items to be picked in a batch. For selecting solutions from a population, the roulette wheel selection approach was used in both *GAs*. Lin et al. (2016) also investigated the joint order batching and order picker routing problem in a single-block warehouse with a single depot. The authors used a modified version of the *PSO* approach originally proposed by Selvakumar and Thanushkodi (2007) for solving the routing problem for a batch. Ho and Tseng (2006) studied order batching in combination with order picker routing and storage assignment in *standard layout A*. For solving the order picker routing problem, a *simulated annealing (SA)* approach was proposed that aimed on improving solutions found by the *largest gap* heuristic. Chen et al. (2015) developed a non-linear mixed-integer optimization model that simultaneously considers three decision problems, namely order batching, batch sequencing, and order picker routing. The objective of the model is to minimize the total tardiness of customer orders. For finding the minimum total travel time and completion time of a batch, an *ACO* approach was used. Ardjmand et al. (2018) proposed a *Lagrangian decomposition (LD)* heuristic combined with *PSO* to solve an order batching, a batch assignment, and an order picker routing problem with multiple order pickers. The objective of their study was to minimize the time required to complete all batches.

Besides the use of meta-heuristics for solving combined planning problems, several works applied meta-heuristics to other complex order picking problems, which are summarized in the following:

Schrotenboer et al. (2017) considered a situation where the order picker has to drop off returned products at their respective storage locations in addition to the picking of items requested by the customer. The authors proposed a *hybrid genetic algorithm (HGA)* to determine the route for a single order picker. The *HGA*, in this context, combines a *GA* with a *local search*. Moreover, they also investigated the case of multiple order pickers subject to congestion by extending the *HGA*, in which order picker interaction is taken into account in the model.

Chabot et al. (2017) used an *adaptive large neighborhood search (ALNS)* (originally proposed by Ropke and Pisinger, 2006) to solve the order picker routing problem with PC. The *ALNS* uses destroy and repair operations to improve the solution in each iteration. A destroy operation removes nodes from the pick sequence, while the repair operation inserts them at potentially better positions. In this study, the authors used three destroy operators, namely the Shaw removal (Shaw, 1997), the worst removal, and a random removal (Ropke and Pisinger, 2006), as well as two repair operators, namely a greedy parallel insertion and a k-regret heuristic (Potvin and Rousseau, 1993). Each operator was selected with a probability based on its past performance, and an acceptance criterion based on a *SA* approach was used for accepting a solution. Bódis and Botzheim (2018) applied a *bacterial memetic algorithm (BMA)* to solve the order picker routing problem based on pallet loading features depending on item properties, pick-list characteristics, and order picking system characteristics. Given a pick-list, storage locations have to be visited and the retrieved items need to be arranged on a pallet in a way that ensures the build-up of a stable transport unit without causing product damages. The pallet setup possibilities and the pick sequences were given in a matrix.

Cortés et al. (2017) studied the order picker routing problem where a tour is generated by simultaneously taking into account product attributes (weight and volume), storage locations (different height levels), inventory availability, and the availability of heterogeneous material handling equipment in the warehouse. The authors applied a generic *TS* and its hybrid variations with *2-Opt Exchange* and *2-Opt Insertion*. The generic *TS* procedure relies on swap and shift movements between two locations to explore a neighboring solution. The former hybrid variant swaps a couple of locations with another couple, whereas the latter variant shifts a couple of locations into a new position within the route.

2.5.1.2 Multi-block warehouses

2.5.1.2.1 Exact algorithm

The exact algorithms based on RR discussed in Section 2.5.1.1.1 are not directly applicable to multi-block warehouses, and have therefore frequently been modified in the past to cover this warehouse layout as well. Roodbergen and de Koster (2001a) studied a conventional warehouse with a middle cross aisle dividing the warehouse into an upper and a lower block. The authors applied the concept of RR to iteratively construct a minimum-length tour subgraph by expanding subgraphs according to the following three transitions: (i) add edges corresponding to possible moves of an order picker within the current aisle in the lower block; (ii) add edges corresponding to possible moves of an order picker within the same aisle in the upper block; and (iii) add edges corresponding to possible moves of an order picker from the current aisle to the adjacent aisle. For transitions (i) and (ii), possible edges presented in RR's work were used. For transition (iii), the authors proposed new edge configurations connecting two adjacent aisles. The time complexity of the algorithm is $O(m+n)$. Roodbergen and de Koster (2001a) assumed that all three cross aisles do not contain any storage locations. Jang and Sun (2012) relaxed this assumption and studied the case where the back cross aisle may contain requested items as well. Since edges corresponding to changeovers from one aisle to the next proposed by Roodbergen and de Koster (2001a) do not cover the case where the back cross aisle contains storage locations, Jang and Sun (2012) proposed additional edges corresponding to possible moves of an order picker within the back cross aisle. They then applied the algorithm of Roodbergen and de Koster (2001a) to find the minimum-length order picking tour. Pansart et al. (2018) applied a fixed-parameter algorithm for the rectilinear TSP discussed in Cambazard and Catusse (2018) to find the optimal route for an order picker through a multi-block warehouse. This algorithm is based on a dynamic programming procedure that defines the states as possible configurations of the separator (degree parity of the vertices and connected components) as well as two types of transitions between states: vertical and horizontal. Horizontal and vertical transitions add vertexes and edges using horizontal and vertical components identified by RR.

Some authors also proposed exact algorithms originally developed for solving the TSP for the order picker routing problem in multi-block warehouses. Roodbergen and de Koster (2001b) applied a *branch-and-bound* method to their TSP formulation to find an order picking tour with minimal travel time in a narrow-aisle warehouse. The drawback of the *branch-and-bound* algorithm is its unpredictable run-time behavior, which would not be suitable for practical implementations. Theys et al. (2010) applied the *exact concorde TSP* algorithm to a conventional warehouse with two blocks and assumed that a picking tour starts and ends at a single depot that can either be in the middle (central depot) or at any other position (decentral depot) in the front cross aisle. The *exact concorde TSP* algorithm was originally developed for solving the symmetric TSP using a *branch-and-cut* method (see Jünger and Naddef, 2001). The *exact concorde TSP* solver (see Applegate et al., 2008) was applied to find the shortest route for a given pick-list. Matusiak et al. (2014) used the (exact) *A*-algorithm*, which is based on dynamic programming, to solve the combined precedence-constrained order picker routing and order batching problem in a multi-block conventional warehouse. The authors assumed that there are multiple depots located at the back cross aisle. Besides the constraint that certain items have to be picked in a pre-specified sequence, the authors assumed that each order has to be delivered to its respective pre-specified depot. Given a number of bins per device, an order picker with an empty device receives a batch of orders where the number of orders contained in the batch is subject to a capacity constraint. An order picker travels through the warehouse to pick the items by separating different customer orders into different bins. Once all orders in a batch have been picked and delivered to their respective pre-specified depots, the order picker receives a new batch of orders. For each batch, the *A*-algorithm* proposed by Hart et al. (1968) was applied to find a picking tour of minimal length. This algorithm uses dynamic programming where a state represents the number of picked items of each order in a batch and the order of the last picked item. The initial state is at the depot when the device is empty, while the final state is when all the items have been picked and delivered, and the device is empty again.

2.5.1.2.2 Heuristics

Routing heuristics for multi-block warehouses can be classified into *simple heuristics* and *improvement heuristics*. With respect to *simple heuristics*, Roodbergen and de Koster (2001b) extended the *largest gap* and the *S-shape* heuristic to the case of a multi-block narrow-aisle warehouse with a single depot located at the front of the left-most picking aisle. In addition, they proposed new routing heuristics they termed *combined* and *combined+*. The proposed routing heuristics can be summarized as follows:

- *Multi-block S-shape:* The order picker starts in the left-most aisle that contains requested items and traverses up to the front cross aisle of the farthest block from the depot that contains requested items. The order picker then moves to the right until s/he reaches a sub-aisle of the farthest block containing requested items. S/he traverses this sub-aisle completely up to the back cross aisle of the farthest block. The picker then moves to either the left- or the right-most sub-aisle containing requested items, depending on which results in the shortest travel distance. S/he then applies the *S-shape* policy for the single-block warehouse to this block and then returns to the front cross aisle of the current block. The picker continues in this fashion to the next block closer to the depot until the last block closest to the depot has been completed. Figure A.3(1) (in the appendix) shows the route resulting from the *multi-block S-shape* heuristic.

- *Multi-block largest gap:* This heuristic uses the same principle for sequencing blocks for picking items as the *multi-block S-shape* heuristic. Once the order picker has reached the farthest block, s/he visits all sub-aisles that need to be entered from the back and then traverses the last sub-aisle completely to the front cross aisle. Note that each sub-aisle that contains items to be picked is entered up to the *largest gap*. The picker then visits all sub-aisles with items left from the front by entering each sub-aisle up to the *largest gap*. The picker continues in this fashion to the other blocks until the last block has been completed (see Figure A.3(2) in the appendix for the route generated by *multi-block largest gap*).
- *Combined:* The order picker starts in the left-most aisle that contains requested items. S/he traverses this aisle up to the front cross aisle of the farthest block that contains requested items. S/he further picks the requested items in the farthest block where the sub-aisles are visited sequentially from left to right. The picker either traverses each sub-aisle completely or enters and leaves the sub-aisle from the same side. This choice is made with the help of a dynamic programming method. The picker continues picking in this fashion until all blocks with requested items have been visited. Figure A.4(1) (in the appendix) shows the route resulting from the *combined* heuristic.
- *Combined+:* This heuristic improves the *combined* heuristic in two ways. First, the sub-aisles containing requested items in the block closest to the depot are visited from right to left. Secondly, the farthest block is not necessarily accessed using the left-most aisle of the warehouse containing requested items. Instead, it can be accessed by the left-most sub-aisle with items of the block closest to the depot. Routes generated by the *combined+* heuristic are at least as good as routes generated by the *combined* heuristic. Figure A.4(2) (in the appendix) presents the route obtained by the *combined+* heuristic.

Chen et al. (2013) proposed two modified *S-shape* heuristics for the case where congestion (picker blocking) can occur. The first one applies the traditional *S-shape* heuristic under the condition that an order picker has to wait at the entrance of an aisle in case it is occupied by another picker until the aisle has been cleared. The second *S-shape*$^+$ heuristic considers three types of spatial relationships between a picked item and the next target item, namely: (i) they are in the same sub-aisle; (ii) they are in the same block, but in different sub-aisles; and (iii) they are in different blocks. The authors used these three relationships to determine the travel time as well as the waiting time when congestion occurs while picking the requested items.

Vaughan and Petersen (1999) developed the *aisle-by-aisle* heuristic. The authors considered the case where an order picking tour starts at the front end of the left-most aisle and ends at the front end of the right-most aisle containing requested items. This heuristic proceeds from the left- to the right-most picking aisle under the condition that each picking aisle containing requested items has to be visited exactly once. A dynamic programming approach was applied to determine the best cross aisle to use for moving from one picking aisle to the next in such a way that the travel distance generated by the proposed heuristic is minimized. Matusiak et al. (2017) proposed a new routing heuristic they termed the *middle aisle multi-drop off routing heuristic*. This heuristic is a combination of the modified *aisle-by-aisle* heuristic (Vaughan and Petersen 1999) and *Dijkstra's algorithm* (Dijkstra, 1959). Once a partial tour for retrieving all orders in a batch has been formed according to the *aisle-by-aisle* heuristic, the overall picking tour is completed by adding the visited depots with *Dijkstra's algorithm*. Their proposed heuristic was used in a joint order batching, picker assignment, and order picker routing problem by taking into account the order pickers' skills in assigning batches to pickers. The authors considered multiple depots at the back cross aisle. Each order is assigned to a pre-specified depot that has to be visited after all items contained in an order have been picked. All picking aisles in the warehouse can only be traversed in a single direction, while the cross aisles can be traversed in both directions.

Scholz and Wäscher (2017) studied the joint order batching and order picker routing problem considering a single depot located at the front of the left-most picking aisle. For routing order pickers through the warehouse, the authors proposed a new heuristic they termed the *heuristically modified exact algorithm*. This algorithm builds on the exact algorithm of Roodbergen and de Koster (2001a) and tries to reduce the number of subgraphs constructed in each iteration by deleting all PTSs, except the shortest one, after each change of a picking aisle.

Shouman et al. (2007) proposed the following two heuristics:

- *Block-aisle 1:* Each block in the warehouse is divided into an upper and a lower part. The upper part consists of storage locations where the distance from the back cross aisle is less than or equal to half of the block length. The rest of the storage locations is assigned to the lower part. The order picker traverses the left-most aisle that contains requested items up to the upper cross aisle of the farthest block that contains requested items. The picker retrieves the items stored in the upper and lower parts using the *return* policy. The same procedure is applied to the other blocks closer to the depot until the last block has been completed. Figure A.5(1) (in the appendix) illustrates a route obtained by the *block-aisle 1* heuristic.
- *Block-aisle 2:* This heuristic is identical to the *Block-aisle 1* heuristic, with the difference that the upper part contains also the next adjacent storage location of the lower part if it

contains the requested item. Figure A.5(2) (in the appendix) illustrates a route resulting from the *block-aisle 2* heuristic.

Chen et al. (2019a) proposed heuristics for the case of ultra-narrow aisles and access restriction, where an order picker cannot enter the sub-aisle with a picking device, but has to leave the device at the aisle entrance. Moreover, for some sub-aisles, the order picker can enter and leave from only one side of the aisle because the other side is blocked by the warehouse wall. The warehouse investigated consists of cross aisles and connect aisles that are perpendicular to the cross aisles (this warehouse layout is illustrated in Figure A.6 in the appendix). The existence of cross and connect aisles divides the warehouse into multiple blocks in the direction from front to back and from left to right. The order picker uses either cross or connect aisles for travelling from the current sub-aisle to the target sub-aisle, where the intersection between a cross aisle and a connect aisle is referred to as a cross point. The authors extended the *return, largest gap,* and *midpoint* heuristics to make them applicable to the investigated warehouse. The proposed heuristics can be summarized as follows:

- *Return for ultra-narrow aisles and access restriction (RNA):* The order picker starts at the depot, moves through the left-most cross point, and accesses the first cross aisle with picking tasks. For each sub-aisle that contains requested items, the order picker selects the shortest feasible pick mode using the *return* policy. Afterwards, the order picker moves through the nearest cross point to access the next cross aisle with picking tasks and continues in this fashion until the last cross aisle has been completed. Figure A.6 presents an example of a route generated by the *RNA* heuristic.
- *Largest gap for ultra-narrow aisles and access restriction (LNA):* This heuristic uses the same principle as the *RNA* heuristic. The difference is that the order picker selects the shortest feasible pick mode using the *largest gap* policy.
- *Midpoint for ultra-narrow aisles and access restriction (MNA):* This heuristic uses the same principle as the *RNA* heuristic. The difference is that the order picker selects the shortest feasible pick mode using the *midpoint* policy.

Given that travelling in a connect aisle is a non-value adding activity, the authors proposed three additional heuristics, namely *return for ultra-narrow aisles and access restriction plus (RNAP)*, *largest gap for ultra-narrow aisles and access restriction plus (LNAP)*, and *midpoint for ultra-narrow aisles and access restriction plus (MNAP).* These three heuristics use the same principle as the *RNA, LNA, MNA*, respectively. The difference is that there is no connect aisle between two adjacent blocks in the left and right direction. Without a connect aisle, the warehouse is divided into multiple blocks only in the front and back direction.

Improvement heuristics try to improve an initial solution generated by a heuristic. Popular improvement heuristics are the *2-opt* and the *3-opt* local searches as well as the *Lin-*

Kernighan-Helsgaun (LKH) TSP heuristic. Hsieh and Tsai (2006) adapted the *Z-pick* heuristic proposed by Goetschalckx and Ratliff (1988a) to a multi-block warehouse by relaxing the limitation that the order picker has to go back and forward along the two sides of an aisle when the pick density within this aisle is high, which may result in an unnecessarily high travel distance. The authors first used the traditional *Z-pick* heuristic to find an initial tour. Afterwards, they applied a *2-opt local search* algorithm originally proposed by Croes (1958) to change the pick sequence in order to obtain a shorter route. For moving from one aisle to the next, the *S-shape* principle was used. In this study, the authors assumed that each picking tour starts at an input point located at the left-most front cross aisle and ends at an output point located at the right-most front cross aisle. Kulak et al. (2012) proposed two heuristics for determining picking tours. First, they combined a *nearest neighbor* heuristic with an *Or-opt* heuristic. For the *nearest neighbor* heuristic, the order picker starts at the depot and then travels to the nearest pick location. From this location, s/he travels to the next (nearest) pick location etc. until all requested items have been retrieved. The *Or-opt* heuristic proposed by Or (1976) was then used to modify this initial tour by removing two or three consecutive pick locations from the picking tour and reinserting them at a different location into the tour. Secondly, the authors combined a *savings algorithm* with the *2-opt* heuristic.

Pferschy and Schauer (2018) proposed three routing heuristics based on insertion methods for determining picking tours, namely *farthest insertion*, *cheapest insertion*, and *random insertion*. These heuristics start with a tour consisting of two nodes and then add the remaining nodes that are not in the tour one by one in the shortest possible way. The initial solutions generated by the heuristics are then improved by applying the *3-opt local search*.

Çelik and Süral (2019) developed an order picker routing heuristic they denoted *merge-and-reach* for a narrow-aisle warehouse with multiple blocks and a single depot at the front of the left-most picking aisle. This heuristic initially divides the warehouse into two parts using a cross aisle. For each part, a route is constructed using the algorithm of RR. The heuristic then checks if solutions overlap by comparing the solutions for two adjacent blocks starting from the lowest to the upper-most block. If the solutions overlap, they are merged by deleting a set of edges from their union without losing connectivity. Otherwise, they are joined by connecting them in the shortest possible way. The solution for the entire warehouse can be found using the same procedure by finally merging the solutions of both warehouse parts. The solution is finally further improved by applying the *3-opt local search*. The time complexity of the proposed heuristic is $O(k^2n^3 + m^2)$, where $k, n,$ and, m represent the number of cross aisles, picking aisles, and requested items, respectively.

Theys et al. (2010) used the *LKH* TSP heuristic (see Lin and Kernighan, 1973 and Helsgaun, 2000) to solve the order picker routing problem in a conventional warehouse with two blocks. The *LKH* is a local optimization algorithm that takes an initial order picking tour and then repeatedly exchanges some edges in the tour with other edges that are not in the tour (based on the $\lambda - opt$ algorithm presented in Lin, 1965) to reduce the distance of the current tour. Numerical experiments showed that *LKH* leads to significant improvements in travel distance as compared to existing heuristics (*multi-block S-shape, multi-block largest gap, combined,* and *aisle-by-aisle*). The authors further used the same heuristics to generate an initial tour that was then improved with the *LKH* heuristic. These combinations slightly improve the travel distance generated by *LKH* heuristic with default initial solution. Furthermore, they also considered the combination of the same heuristics with a *2-opt local search* heuristic. This way, they generated four routing heuristics, namely *multi-block S-shape + 2-opt*, *multi-block largest gap + 2-opt*, *combined + 2-opt*, and *aisle-by-aisle + 2-opt*. These heuristics improved the initial solutions generated by those heuristics without substantial increases in run time.

Scholz et al. (2017) investigated a narrow-aisle warehouse with two blocks and a single depot located at the front of the left-most picking aisle. They used the *combined* heuristic to generate initial tours, which then improved using the *LKH* as well as the *2-opt* and *3-opt* heuristics.

2.5.1.2.3 Meta-heuristics

Meta-heuristics for solving the order picker routing problem have also been proposed for multi-block warehouses, and in many cases, they are based on ACO. Also in this case, they were mainly used for solving combined planning problem as well as order picker routing problems in complex scenarios. Chen et al. (2013) (discussed in Section 2.5.1.2.2) also applied an *ACO* approach to a multi-block narrow-aisle warehouse with two order pickers taking into account congestion. The authors represented the routing problem as a Steiner TSP, which can be solved using *ACO*. To deal with the congestion problem, they proposed some spatial relationship between a picked item and the next target item. This work was extended by Chen et al. (2016), who proposed a routing method based on *ACO* for multiple order pickers under stochastic picking times and order picker congestion. The authors first determined an initial route for each order picker by applying the *ACO* meta-heuristic proposed by Chen et al. (2013), and then coordinated the routes of the order pickers in real time by online coordination rules that can be used to inspect the other order pickers' positions. Li et al. (2017) used an *ACO* approach combined with a *2-opt* local search for solving the joint order batching and order picker routing problem in a warehouse with two blocks and a single depot. First, a feasible picking tour is constructed using *ACO*. After that, two variants of a local search procedure, namely *2-opt reverse* and *2-opt relocate* as proposed in Zhang et al. (2013), are applied to improve the initial solution. Both local search procedures choose the two nodes *X* and *Y* from the current tour. The *2-opt reverse* procedure reverses the partial sequence from *X* to *Y*, whereas the *2-opt relocate* procedure inserts *Y* in front of *X*.

De Santis et al. (2018) proposed a new hybrid meta-heuristic, *ACO* in combination with *Floyd-Warshall (FW),* for order picker routing in a narrow-aisle warehouse with two blocks and a single depot in a low-level single storage warehouse. The picking tour starts and ends at the depot. In the first stage of this hybrid heuristic, a graph representation of the warehouse is constructed, and then the *FW* algorithm (Floyd, 1962; Warshall, 1962) is used to find the shortest path connecting each pair of vertices in the graph, where the input to the *FW* algorithm is the graph representation. In the second stage of the procedure, the *ACO* algorithm determines the picking route. The proposed algorithm is applicable not only for static order picking, but also for dynamic order picking. Furthermore, the authors used the *MAX-MIN ant system (MMAS)* algorithm (Stützle and Hoos, 2000) as a benchmark to evaluate the performance of the *FW-ACO* algorithm. Recently, Chen et al. (2019b) developed a hybrid of an *ACO* and a *GA* for order picking in the multi-block warehouse with ultra-narrow aisles and access restriction, which is a layout similar to that investigated in Chen et al. (2019a). Their algorithm uses *ACO* for generating the initial chromosomes for the *GA*, and the *GA* is then used for determining the route. The authors compared their hybrid meta-heuristics with the *RNA* and *LNA* heuristics proposed by Chen et al. (2019a), and found that their proposed method outperforms such heuristics in most investigated scenarios. Besides routing meta-heuristics based on *ACO* and its hybrid variants, Lin et al. (2016) proposed a *PSO* procedure for solving the order picker routing problem in a multi-block warehouse. The reader is referred to Lin et al. (2016) for details (see Section 2.5.1.1.3).

2.5.2 Non-conventional warehouse

2.5.2.1 Exact algorithm

Exact routing algorithms for non-conventional warehouses have rarely been proposed so far. Çelik and Süral (2014) investigated the so-called fishbone warehouse, where picking aisles extend horizontally and vertically from two diagonal cross aisles. The authors proposed a tractable transformation from a graph representation of the fishbone warehouse to a graph for a conventional warehouse with two blocks presented in Roodbergen and de Koster (2001a). The authors stated that this transformation is applicable also to the flying-V warehouse, which consists of a middle cross aisle aligned in a V-shape, and parallel picking aisles that are perpendicular to the front and back cross aisles (see, e.g., Gue and Meller, 2009). Consequently, the order picker routing problem can be solved optimally for both fishbone and flying-V warehouses in polynomial time using both the transformation and the algorithm proposed in Roodbergen and de Koster (2001a). Öztürkoğlu and Hoser (2019) also used the algorithm of RR and modified the algorithm of Roodbergen and de Koster (2001a) for optimally routing order pickers through a new warehouse design called a discrete cross aisle layout (see Figure A.7 in the appendix). In this layout, a traditional middle cross aisle is divided into segments called tunnels, where each tunnel connects two adjacent pick aisles. The authors developed an efficient procedure for constructing PTSs for alternative positions of a tunnel for each aisle.

Glock et al. (2019) recently integrated human factors aspects into a model for rotating pallets in a non-conventional warehouse with a U-shape picking zone with a single depot located at the open end of the zone. The authors proposed an optimal routing policy by utilizing the fact that all requested items are located on a convex polygon. Based on the theorem of Barachet (1957), a picking tour starts at the depot and then continues to the requested items in clockwise order from the depot, ending at the depot. All requested items on a pick-list are picked according to the pick-by-order principle operated by a single order picker.

2.5.2.2 Heuristics

Besides the exact algorithm presented in the previous section, Çelik and Süral (2014) also modified several simple heuristics to make them applicable to the fishbone warehouse, namely *S-shape*, *largest gap*, and *aisle-by-aisle*. The fishbone warehouse is divided by two diagonal middle aisles into three parts, referred to as the left, the middle, and the right parts. The modified heuristics can be summarized as follows:

- *Fishbone S-shape*: The order picker starts in the left part of the warehouse and applies the *S-shape* heuristic from the first aisle that contains requested items to the back-most aisle. S/he then moves to the middle part of the warehouse and picks requested items according to the same principle, and finally completes the right part. The middle part is completed from left to right, and the right part from the back to the front.
- *Fishbone largest gap*: This heuristic first applies the *largest gap* heuristic to the left and middle parts of the warehouse using the left diagonal middle aisle. Afterwards, the order picker moves to the back cross aisle of the middle part and picks items from this part. S/he then moves to the right diagonal middle aisle and picks the remaining items in the middle part of the warehouse as well as items in the right part, following the *largest gap* heuristic. Next, the order picker moves back to the depot and thereby picks all remaining items.
- *Fishbone aisle-by-aisle*: This heuristic proceeds sequentially from the left to the right part of the warehouse and applies the *aisle-by-aisle* strategy to each part.

Henn et al. (2013) developed a heuristic routing procedure for a U-shaped layout that was presented earlier in Gerking (2009). The U-shaped layout consists of a central aisle arranged in the form of a U, with various picking aisles extending from the central aisle. The central and front cross aisles are wide aisles that allow order pickers to pass each other, while all other aisles are narrow. Narrow aisles cannot be entered with a picking device. As a result, the order picker only travels on the central aisle with his/her device and enters the actual picking aisles without the device. This entails that a picking aisle may have to be visited more than once if multiple items are requested from this aisle, which was referred to as the *return-with-replication* policy (see also Kunder and Gudehus, 1975). If a requested item is stored in the central block, the order picker can enter the aisle either from the right or from the left of the central aisle, depending on which option gives the shortest route. During walking in the central aisle, the order picker has to decide about whether to pick the requested items from the left or from the right of the central aisle since only one-way traffic is allowed in the central aisle. The corresponding routing heuristic was referred to as *walking-the-U*.

Glock and Grosse (2012) also studied a U-shaped warehouse with two parallel shelves and a third shelf that is perpendicular to the two parallel shelves and considered the layout design, storage assignment and order picker routing problems in this case. The authors assumed that the order picker has a limited transport capacity, such that an order may have to be split up into multiple tours. This variant of the capacitated vehicle routing problem was solved using a *sweep algorithm*.

2.5.2.3 Meta-heuristics

Recently, Zhou et al. (2019) developed three routing meta-heuristics, namely a *GA*, an *ACO* approach, and a *cuckoo* algorithm to solve the order picker routing problem in non-conventional fishbone warehouses with narrow aisles and a single storage system. The authors compared the performance of these three algorithms in terms of average tour length and computing time and found that (based on their analysis) the *cuckoo* algorithm is better than *ACO* and *ACO* is better than the *GA*.

2.5.3 General warehouse

Singh and van Oudheusden (1997) studied the case of a warehouse with scattered storage. The authors did not consider a particular warehouse layout, but instead formulated the problem as a variant of the travelling purchaser problem, where the objective is to find a tour that minimizes the sum of travelling and commodity cost. The authors presented a *branch-and-bound* algorithm for this problem that works for any kind of distance matrix that considers travel distances from one storage location to another.

Daniels et al. (1998) also studied the order picker routing problem with scattered storage without assuming a specific layout. The author applied modified *nearest neighbor* and *shortest arc* heuristics as well as a *TS* approach to solve the problem.

Recently, Ardjmand et al. (2019) investigated the order batching and order picker routing problem in a put wall-based order picking systems. Two *GAs* with random shuffling and inverse-insert-swap mutation operations, a *list-based simulated annealing (LBSA)*, and a hybrid of a *GA* and an *LBSA* were proposed to solve this problem. The warehouse investigated in their study is a general warehouse with a single storage system. A picking tour starts at the put wall, continues through the warehouse for retrieving requested items, and returns to the put wall to position each retrieved item in a specific put wall container.

2.6 Discussion

2.6.1 Frequencies of usage

This section analyzes how frequently the routing policies contained in the core sample have been used in the literature. For this purpose, we analyze the frequencies of usage of these policies in all 203 sampled papers (core and extended sample).

As can be seen in Figure 2.6, heuristics have enjoyed the highest popularity in the literature, accounting for 77.0% of all sampled papers. Even though heuristics usually do not generate an optimal route and their gaps to the optimal solutions can be large at times, they have widely been used as they are easy to implement and easy to understand by the order pickers. In addition, some authors mentioned that heuristics generate tours that are intuitive to the order picker, which may be another reason for their popularity (e.g., Petersen et al., 2004). Finally, they are often easier to adapt to alternative layouts, whereas exact algorithms are often dedicated to specific warehouse layouts. Exact algorithms were applied in 13.8% of the sampled papers. Some researchers have noted that exact algorithms are only infrequently used in practice because optimal routes may seem illogical to the order pickers (e.g., de Koster et al., 2007), which may confuse the order picker, inducing deviations from the route (e.g., Petersen and Aase, 2004; Elbert et al., 2017). This could be one reason for the comparatively low popularity of exact algorithms in the literature. Meta-heuristics have also not attracted much attention in the literature so far. As can be seen in Figure 2.6, they account for only 9.2% of the sampled papers. However, the use of meta-heuristics has recently increased (see Figure 2.8), and they could become more popular in the future.

A more detailed analysis of the algorithm categories shows that the *S-shape* heuristic is by far the most popular heuristic, followed by the *return* and *largest gap* policies, which account for 27.7%, 12.2%, and 10.7% of the sampled papers using a heuristic, respectively. According to the authors' experience, the *S-shape* policy is frequently used in practice because of its simplicity, which could be one reason for its popularity in the literature. In terms of exact algorithms, the top three most popular policies are those of RR, its modifications (e.g., de Koster and van der Poort, 1998; Roodbergen and de Koster, 2001a), and the *branch-and-bound* procedure, which account for 42.4%, 25.4%, and 6.8% of the sampled papers applying an exact algorithm, respectively. The popularity of RR's algorithm and its modifications is mainly due to its low run-time, which enables warehouse managers to compute optimal order picking routes quickly. The algorithm of RR and its modifications can solve any realistically-sized problem within fractions of seconds, which is not the case for standard TSP algorithms (Scholz et al., 2016). *Branch-and-bound* algorithms have also been used to find order picking tours with minimal length. Their run-time, however, often prohibits their use in practice. As to meta-heuristics, the three most popular algorithms are *genetic algorithms*, *ant colony optimization*, and *largest gap combined with simulated annealing*, which account for 23.1%, 15.4%, and 7.7% of the sampled papers applying a meta-heuristic, respectively. These heuristics enjoyed an especially high popularity for complex order picking problems that are difficult to solve (van Gils et al., 2018), e.g. joint order batching and order picker routing problems such as the one discussed in Li et al. (2017).

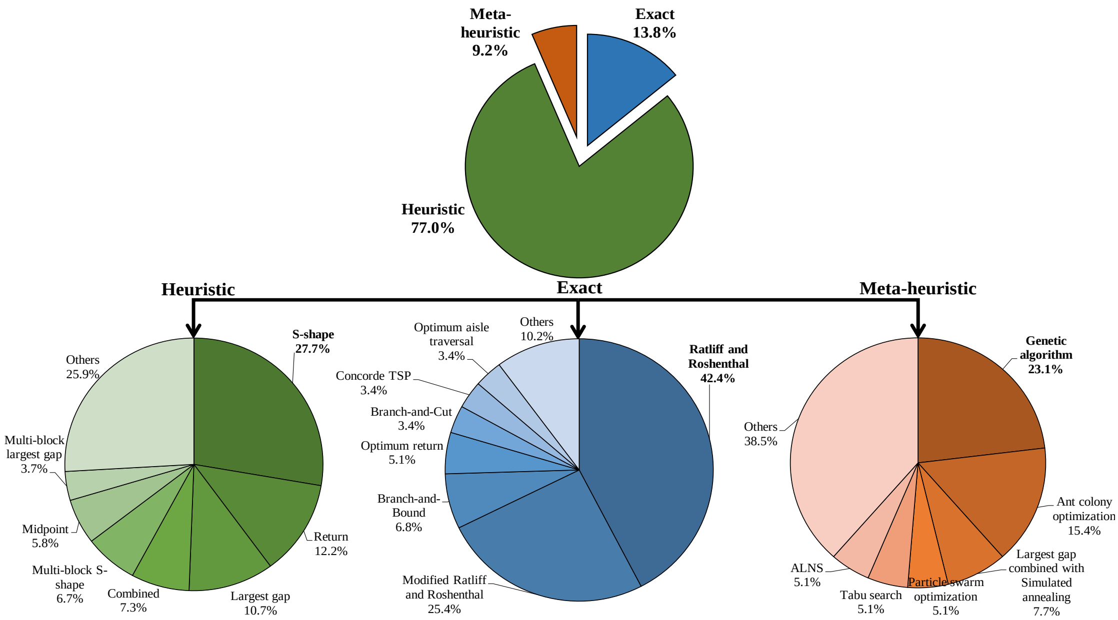


Figure 2.6 Frequencies of usage of the different routing policies.

2.6.2 Citation analysis

A citation analysis can be used to illustrate the connectivity between papers in a literature sample and to identify works that have been pivotal for shaping a specific research field (e.g., Fahimnia et al., 2015). **Figure 2.7** shows the citation graph obtained for the papers in our sample using the Fruchterman Reingold layout in Gephi (https://gephi.org/). The nodes in the graph represent the papers in the core and extended samples, and the edges represent the local citations among them. The size of the nodes reflects the number of local citations a paper received within our sample. Nodes are categorized by different colors according to the main type of algorithm developed/used in the paper (i.e., red: exact; green: heuristic; blue: meta-heuristic). As can be seen, two papers adopt a key position in order picker routing: 1) Ratliff and Rosenthal (1983) and 2) Hall (1993) (marked with numbers 1 and 2 in the graph), receiving 94 and 84 local citations in our sample, respectively. **Figure A.8** in the appendix further illustrates the citation network of these two key papers. As can be seen, these two papers

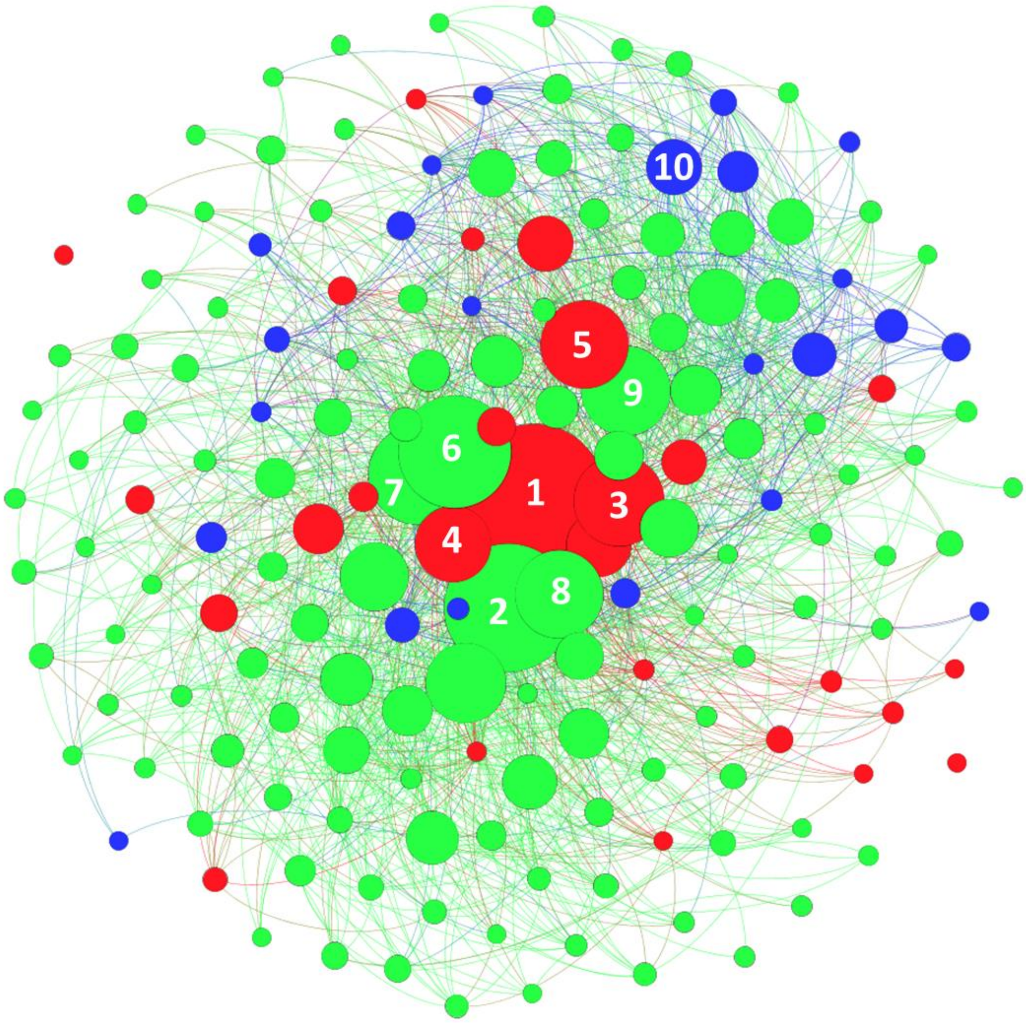


Figure 2.7 Citation analysis of the papers in the core and extended sample.

inspired various works on order picker routing, and they have been relevant for all three types of algorithms discussed in this review. Other papers that contributed especially towards the development and application of exact routing algorithms that received ample citations are 3) Roodbergen and de Koster (2001a), 4) de Koster and van der Poort (1998), and 5) Gademann and van de Velde (2005). With regard to the development of heuristics, the papers of 6) Roodbergen and de Koster (2001b), 7) Petersen (1997), 8) Petersen and Schmenner (1999), and 9) de Koster et al. (1999) are the most cited works. The paper of Roodbergen and de Koster (2001b) is at the center of our citation graph, which illustrates that it is connected to the three algorithm categories in a quite balanced way (cf. **Figure A.8**). The most cited paper proposing a meta-heuristic for order picker routing is 10) Tsai et al. (2008). **Figure A.8** shows that this paper is connected especially to other works proposing meta-heuristics and to papers that propose or apply heuristics. **Figure A.9** illustrates the most contributing authors who published works contained in our core and extended samples, and **Figure A.10** highlights their collaboration structure (again developed using Gephi). As can be seen, there are few main clusters of authors who frequently published together in a specific sub-area of order picker routing. Three important clusters are authors around de Koster and Roodbergen especially for exact algorithms, Petersen and co-authors for heuristic algorithms, and Chen and colleagues for meta-heuristics (see **Figure A.11-Figure A.13**). Finally, **Figure A.14** shows the most cited papers contained in our core sample according to their citations in Google Scholar to highlight the attention the core journal papers received also outside of our (core and extended) sample.

2.6.3 Main insights

Table A.1 in the appendix classifies all papers contained in the core sample in light of the conceptual framework, including warehouse layout, warehouse operations, and algorithm characteristics. As can be seen, the majority of the proposed algorithms focused on conventional, rectangular warehouses, accounting for 83% (45 out of 54) of the sampled works. The most frequently discussed conventional warehouses were single-block warehouses (53%; 24 out of 45 papers), which could be a result of their high level of space utilization. Furthermore, the majority of routing algorithms were developed for narrow-aisle warehouses with a single depot and low-level storage racks. In terms of warehouse operations, most of the proposed routing algorithms were confined to a single order picker and a single storage system without considering any interdependencies, e.g. picker blocking. The studies of Chen et al. (2013; 2016) and Schrotenboer et al. (2017) are the only three studies that proposed routing algorithms considering picker blocking in warehouses. Moreover, most of the studies focused on static picking systems, and only two studies developed routing algorithms for dynamic picking systems (Lu et al. 2016 and de Santis et al. 2018). With respect to order picker routing with precedence constraints, only four papers considered this scenario.

Algorithms for routing order pickers through a warehouse were assigned to three categories in this review, namely exact algorithms, heuristics, and meta-heuristics. Table A.1 shows that 18 out of 61 algorithms we identified are exact algorithms, 26 are heuristics, and 17 are meta-heuristics.

Figure 2.8 shows that all three types of routing policies have enjoyed an increasing popularity over the years; we could, however, not identify a trend that indicates that one type has become (much) more popular than the two others over time. Exact algorithms that exploit the special distance matrices that occur in warehouses have frequently been proposed in the past. They are mainly based on the algorithm proposed by RR. The most common features of the exact algorithms based on RR are that they first construct a graph representation, generate

Eulerian subgraphs, and then define PTSs. They do not consider each PTS separately, but group PTSs according to their equivalence classes.

Unlike exact algorithms, heuristics were proposed for approximating solutions, and they are often easier to implement. The results of our review showed that two types of routing heuristics were proposed: (i) *constructive heuristics* and (ii) *improvement heuristics*. The first category can be further divided into *simple heuristics* (e.g., *S-shape*, *largest gap*, *composite*) and *TSP heuristics* (e.g., *LKH*, *nearest neighbor*, *savings algorithm*). *Simple heuristics* are simple 'rules of thumb' that can be used for generating straightforward and easy-to-memorize routes. A *simple heuristic* continuously searches for solutions and stops when a solution is found. *Improvement heuristics* usually employ a hybrid method that tries to improve an initial solution that has been generated by either a *simple heuristic* or a *TSP heuristic*. Our results revealed that the *2-opt* and *3-opt* local searches have been frequently used to improve initial tours. Even though routing heuristics have attracted the attention of researchers due to their short run time and their applicability, the biggest drawback of routing heuristics is that their optimality gaps can be large at times. Results reported in the literature show that exact algorithms obtain tour lengths that are between 4% and 18% (Goetschalckx and Ratliff, 1988a), 7% and 34% (de Koster and van der Poort, 1998), 1% and 25% (Roodbergen and de Koster, 2001b), 24.3% (Jang and Sun, 2012), 9% and 38% (Çelik and Süral, 2014), 12% (Lu et al., 2016) shorter than those generated by heuristics. The optimality gaps of the heuristics depend on several factors such as warehouse layouts, warehouse sizes, pick-list sizes, and the solution of other order picking planning problems. For example, Çelik and Süral (2014) reported that the optimality gaps of heuristics investigated in their work decrease when the depth/width ratio of the investigated warehouses increases.

Meta-heuristics have especially been used to solve order picker routing problems that were studied in combination with other planning problems (e.g., batching, storage assignment). The *GA* is the most popular meta-heuristic for order picker routing. One of the most important decisions when implementing a *GA* is to decide on the solution representation. In order picker routing, we found that the most commonly used solution representation of the *GA* is one where the value of a gene denotes the storage location of a requested item, and the order of the genes in a chromosome represents the visiting sequence of the storage locations. Moreover, we found that the most frequently used crossover operator is the *partially matched crossover*.

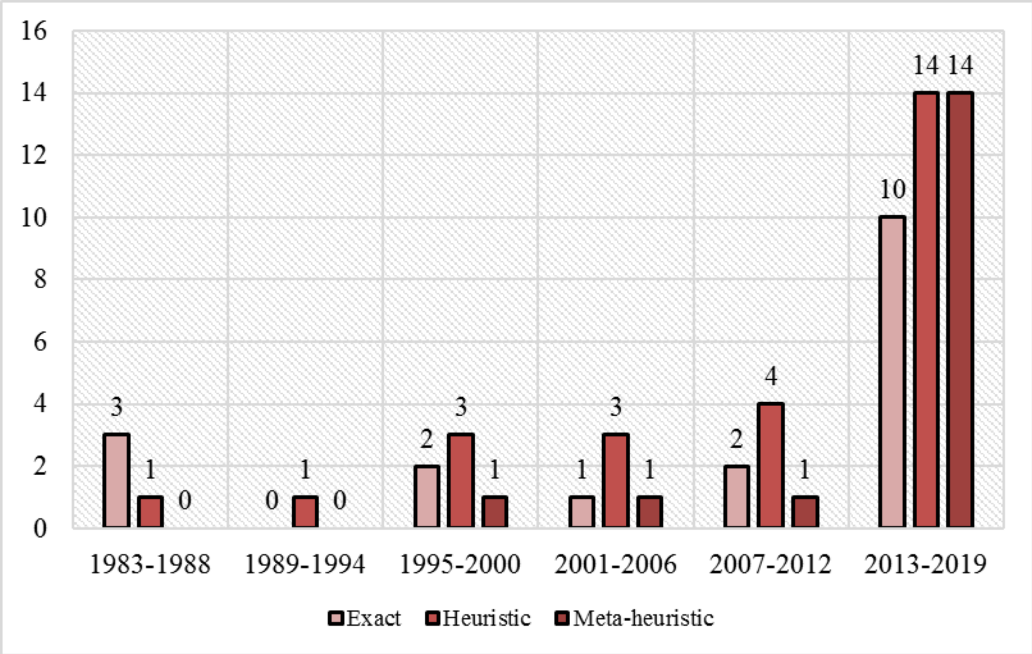


Figure 2.8 Number of order picker routing policies in the core sample per year of publication.

2.6.4 Research opportunities

From the analysis of the papers contained in the core and extended samples, we identified various research opportunities for further developing the research field of order picker routing in warehouses. We categorize the research opportunities with respect to warehouse layouts and warehouse operations. For each category, we formulate research opportunities by priority and relevance to practice.

With respect to *warehouse layouts*, our first observation is that most papers that proposed exact algorithms focused on (conventional or non-conventional) warehouses with a single depot. In practice, however, warehouses may have multiple depots (cf. de Koster and van der Poort, 1998; Matusiak et al., 2014). Therefore, future research could generalize the existing exact algorithms to warehouses with multiple depots. A second observation is that the majority of the proposed exact algorithms were dedicated to the conventional warehouse (see Table A.1). In contrast, we found only three papers that proposed exact algorithms for solving the order picker routing problem in non-conventional warehouses, namely the fishbone and the flying-V (cf. Çelik and Süral, 2014), the U-shaped picking zone (cf. Glock et al., 2019), and the discrete cross aisle layout (cf. Öztürkoğlu and Hoser, 2019). Consequently, there is a strong need for developing exact routing algorithms for other non-conventional warehouses such as other U-shaped layouts (Henn et al., 2013), the inverted-V (Gue et al., 2012), the chevron, the leaf, and the butterfly layouts (Öztürkoğlu et al., 2012). A third observation is that order picker routing policies for the fishbone warehouse have not received much attention by researchers so far, despite the existing evidence that this layout is used in practice (Öztürkoğlu et al., 2012). Hence, developing additional order picker routing policies for the fishbone warehouse could be an interesting topic for future research. A fourth observation is that prior studies that proposed exact routing algorithms for an entire warehouse (in contrast to single aisle only) have almost consistently assumed that the order picker can reach the requested items from both sides of the aisle without having to cross to the other side of the aisle. To fill this research gap, future research could study optimal routing for (entire) wide-aisle warehouses where additional horizon travels within an aisle are taken account of. Another observation we made is that works proposing exact algorithms studied warehouses with either purely wide or purely narrow aisles. We only identified a single paper that used the *S-shape* heuristic to estimate travel time in a warehouse with both wide and narrow aisles (Mowrey and Parikh, 2014). Therefore, it would be interesting to develop exact algorithms for order picker routing in a mixed-width aisle warehouse. We also noticed that there is no exact algorithm for order picker routing in warehouses with access restrictions as described in Chen et al. (2019a,b), which would be another interesting research opportunity.

With respect to *order picking operations*, almost all routing algorithms found in this review focused on a single storage system. Only 4 out of 54 papers in the core sample addressed the routing problem in a scattered storage system (Singh and van Oudheusden, 1997; Daniels et al., 1998; Weidinger, 2018; Weidinger et al., 2019). Therefore, there may be opportunities for developing routing algorithms for this area as scattered storage systems are applied in many real-world warehouses (see Weidinger, 2018). A further topic that has only attracted little attention so far is the routing of order pickers subject to precedence constraints, e.g. based on item weight or item category (food/non-food) (see Chabot et al., 2017; Žulj et al., 2018). According to our experience, the current state-of-research does not reflect the importance precedence constraints enjoy in practice, and therefore we recommend the order picker routing problem with precedence constraints for future research.

The number of papers focusing on the combination of multiple order picking planning problems has increased over the last decade. Solving combined planning problems can lead to an improved warehouse performance (see van Gils et al., 2018). Consequently, future research could continue to investigate the interaction between multiple problems, e.g. order picker routing and batching, order picker routing and storage assignment etc. Since the resulting problems are usually very complex, meta-heuristics could be promising solution approaches.

Several warehouses of online retailers apply dynamic order picking (Gong and de Koster, 2008). However, the results of our review show that dynamic order picker routing is another topic that has not attracted much attention so far. In a dynamic environment, pick-lists can be updated while the order picking process is in progress due to incoming orders that are added to the current tour (e.g., Lu et al, 2016). Our review showed that the optimal routing of order pickers in dynamic environments has thus far only been addressed by Lu et al. (2016) for a single-block warehouse. Therefore, future research could develop exact routing algorithms for situations where items are dynamically added to existing tours. The algorithm could then find a new optimal route that starts at the current position of the order picker and ends at the depot.

Our review also showed that an exact routing algorithm that accounts for picker congestion was not proposed so far. Therefore, future research could focus on developing exact routing algorithms for the case where congestion may occur within aisles. Once congestion occurs, new optimal routes would have to be calculated for all order pickers involved in the congestion. One possible way to approach this problem is to apply some dedicated rules according to the spatial relationship between a picked item and a next target item as proposed by Chen at al. (2013).

Another observation is that Çelik and Süral (2016) were the only to investigate turn penalties that take into account the time that is lost when the order picker changes the direction of travel. Future research could extend their work to other warehouse layouts, e.g. multi-block warehouses or non-conventional warehouses.

Finally, Grosse et al. (2017) pointed out that order picker routing interacts with human factors aspects, such as fatigue, learning or injury risks. Our review showed, however, that human factors aspects have so far only been considered very infrequently (only 1 out of 54 papers in the core sample). Future research could hence propose routing algorithms that take into account the interaction between order picker routing and human factors aspects.

2.7 Conclusion

Order picker routing in warehouses has become an important planning task in every manual order picking system. Travelling through the warehouse for retrieving requested items from storage locations consumes a significant amount of an order picker's working time. To reduce travel time, various order picker routing policies have been proposed in the literature over the last decades. To map the research field of order picker routing and to classify all existing algorithms and warehouse-specific routing procedures, this chapter conducted a systematic review of the literature on order picker routing problems. A conceptual framework was proposed for classifying the different routing policies that have emerged in the literature. Using this framework, we categorized the existing literature with regard to the type of algorithm (exact, heuristic, and meta-heuristic) and warehouse layout (conventional, non-conventional, and general). We provided a structured discussion of the existing routing algorithms following the conceptual framework. We conclude that research on order picker routing in warehouses has received much attention especially over the last five years, where 63.0% of the core sample papers and 55.0% of the extended sample papers have been published. This increasing trend may be an indicator of the importance of order picker routing both in research and industry, despite the automation efforts that are currently made in many industries. We also note that algorithms employed for solving the order picker routing problem differ in terms of their accuracy and computational complexity. Heuristics (77.0% of all sampled papers) have enjoyed the highest popularity in the literature, whereas exact algorithms (13.8% of all sampled papers) have received less attention. Meta-heuristics (9.2% of all sampled papers) have enjoyed the highest popularity for solving combined order picking planning problems that are difficult to solve. Our review shows that the majority of the proposed algorithms focused on conventional warehouses. In contrast, only 6 out of 54 papers contained in the core sample addressed the order picker routing problem in non-conventional warehouses (cf. Glock and Grosse, 2012; Henn et al., 2013; Çelik and Süral, 2014; Glock et al., 2019; Öztürkoğlu and Hoser, 2019; Zhou et al., 2019). For several non-conventional warehouses, there is further potential for developing exact, heuristic and meta-heuristic routing policies.

Our discussion of the state-of-knowledge of order picker routing shows that there is potential for future research to develop exact algorithms for the routing of order pickers, both for non-conventional warehouses and/or for order picking in specific scenarios, e.g. under dynamic picking, picker congestion, turn penalties, or precedence constraints.

Our review at hand has limitations. We only considered papers relevant for this study that were published in peer-reviewed journals, whereas papers that appeared in other outlets (e.g. book chapters or conference proceedings) were excluded from the review. These filters may have led to the exclusion of relevant work from this review. In addition, besides some (anecdotal) evidence found in the reviewed papers, we were not able to report how frequently the different routing policies and warehouse layout types are used in practice. Future research could therefore extend the scope of this review to derive additional insights into the practical use of order picker routing policies and their implementation in warehouse management software. Moreover, our review studied order picker routing for manual picker-to-parts systems. We did not consider the routing of robots in automated warehouses. The routing of robots may differ from the routing of order pickers, e.g. due to a limited battery capacity or constraints on human-robot-interaction. Future research could further investigate the routing of robots to gain further insights into routing problems in warehousing.

Chapter 3 Order picker routing with arbitrary starting and ending points of a tour[5]

3.1 Introduction

The routing of order pickers through the warehouse is an important decision problem because of its strong impact on the total warehouse operating costs. The results obtained in Chapter 2 showed that prior research proposed a variety of routing policies that help managers in improving order picking operations and in reducing the total cost of warehousing. The results of the review of order picker routing policies also revealed that prior research assumed in many cases that each order picking tour starts and ends at the same location, namely the depot. Examples include the works of RR, Roodbergen and de Koster (2001a, b), and Scholz et al. (2016). In practice, however, individual order picking tours do not necessarily start and end at the depot. De Koster and van der Poort (1998), for instance, discussed a scenario where the order pickers have the opportunity to drop off retrieved items at multiple drop-off locations at the head of each aisle. Another example is the case where order picking tours are updated in real time. In this case, while the order picker is travelling through the warehouse, a new order arrives at the warehouse. The warehouse manager or warehouse system then decides to update the order the picker is currently working on, such that a new tour starts at the position of the order picker where the tour update was received. The ending point of the tour would in such a case still be the depot. A paper that investigated this scenario is Lu et al. (2016). Löffler et al. (2018) described another scenario where the order picker is accompanied by an AGV, which automatically drives back to the depot once an order has been completed. The order picker can then continue with the next tour and a new AGV without returning to the depot. The starting point of a tour would then be either the depot or the location of the last pick in the previous tour, while the ending point would either be the depot or the location of last pick of the current tour (from where the AGV starts driving back to the depot). The starting and ending points of a tour in this case can then be any locations in the warehouse. The work of Löffler et al. (2018) proposes a generalization of the models developed by de Koster and van der Poort (1998) and Lu et al. (2016). However, it is only applicable to conventional single-block warehouses. The optimal order picker routing policy with arbitrary starting and ending points of a tour for the two-block warehouse has not been investigated so far. This chapter aims to close this research gap by extending the work of Löffler et al. (2018) to the two-block warehouse. We apply the

[5] Chapter 3 is based on the following working paper: Masae, M., Glock, C.H., Vichitkunakorn, P., Optimal picker routing in a conventional warehouse with two blocks and arbitrary starting and ending points of a tour.

concepts of RR and Roodbergen and de Koster (2001a) that used graph theory and a dynamic programming procedure to find an optimal order picker route in this new scenario. We also propose a routing heuristic, denoted *S*-shape*, for solving the order picker routing problem in this scenario. The main contributions of this study are therefore to (i) develop an exact algorithm and a routing heuristic for a conventional warehouse with two blocks where the starting and ending points of a tour can be any location in the warehouse, (ii) assess the impact of the middle cross aisle on the performance of the warehouse using the average tour length obtained by the exact algorithm, and (iii) evaluate the performance of the proposed heuristic compared to the exact algorithm.

The remainder of this chapter is organized as follows. Section 3.2 describes the problem investigated in this chapter. Section 3.3 introduces and analyzes the optimal routing of order pickers when starting and ending points of a tour are arbitrary. A numerical example is then presented in Section 3.4. Section 3.5 presents extensive computational experiments. Finally, Section 3.6 concludes the chapter and presents an outlook on future research opportunities.

3.2 Problem description and graph representation

The order picker routing problem considered here assumes a conventional warehouse with two blocks, referred to as the upper and the lower blocks, where each block has r parallel aisles with equal length and width. There are three cross aisles, namely the front, the middle, and the back cross aisles, perpendicular to those parallel aisles. The starting and ending points of the desired order picking tour can be any location in the warehouse, and each order picking tour is completed by a single order picker. **Figure 3.1** presents an example of such a conventional warehouse with two blocks, where the black boxes represent the locations of items to be picked, while the two boxes marked with the symbols e_1 and e_2 indicate the starting and ending points of a picking tour, respectively, which are different locations in this example. In a situation where there are many orders and where tours are updated in real-time (i.e., in a dynamic order picking system), the starting point of a tour would be either the position where the pick-list was updated (Lu et al., 2016) or the location of the last pick of the previous tour (Löffler et al., 2018), whereas the ending point would be either the depot (Lu et al., 2016) or the location of last pick of the current tour (Löffler et al., 2018). Note that the proposed procedure can also be applied to the case where the starting and ending points of a tour are the same location (e.g., the depot). All aisles are assumed to be narrow with items stored on the racks on both sides, such that the picker can retrieve the requested items from both sides of an aisle without having to cross the aisle. The chapter at hand also assumes that order pickers working in the same aisle can pass each other, which means that we do not consider order picker congestion within aisles. Moreover, the items can be picked directly from the racks without requiring vertical travels. Such systems have been referred to as low-level picker-to-parts systems.

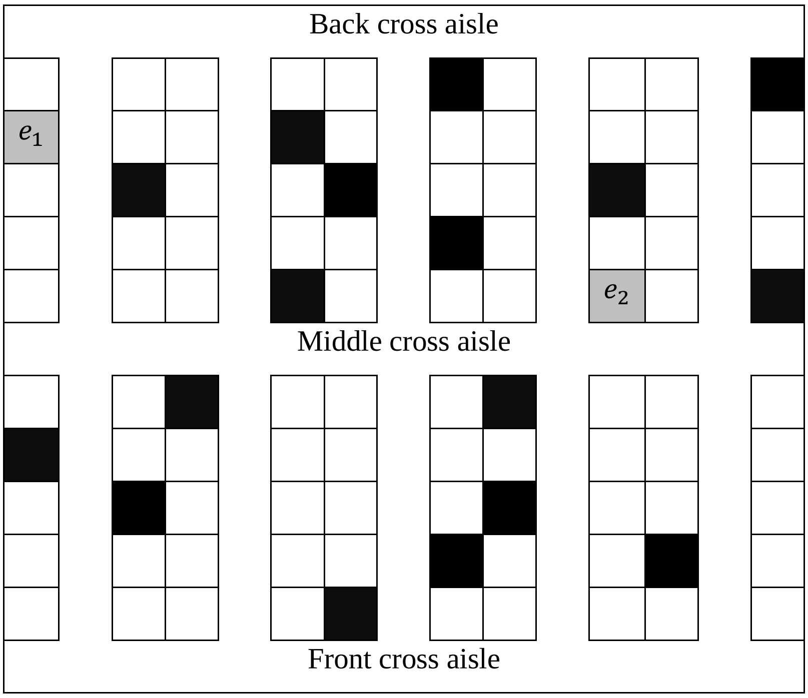


Figure 3.1 Conventional warehouse with two blocks.

Input to the proposed algorithm is a list of locations of m items stored in the warehouse that need to be picked as well as a designated starting point e_1 and a designated ending point e_2 of the desired tour, which can be any position in the warehouse. We note that interchanging e_1 and e_2 gives an equivalent order picking problem as we can reverse the optimal tour. We assume that all m items are separately stored in different shelf locations. There are m_u and m_l items in the upper and lower blocks, respectively, such that $m = m_u + m_l$. The proposed algorithm aims to find the shortest order picking tour that starts from e_1, visits all m storage locations, and ends at e_2. Our algorithm can easily be explained in terms of tours on a graph. We define a graph representation G of our order picking problem by associating vertices e_1 and e_2 with the starting and ending points, and vertices $v_{u,i}$, $i = 1,2, \dots, m_u$ and $v_{l,i}$, $i = 1,2, \dots, m_l$ with the storage locations of the items stored in upper and lower blocks of the warehouse, respectively. For $j = 1,2, \dots, r$, the vertices a_j, b_j, and c_j represent the rear end, the middle end, and the front end of aisle j, respectively. Assigned to each pair of adjacent vertices in the warehouse is an undirected and infinite number of parallel edges. However, from Theorem 3.3 in Section 3.3.2, no more than two edges between each pair of adjacent vertices are contained in the minimum-length order picking tour subgraph. Consequently, we can assume that each pair of adjacent vertices in the warehouse is connected by two parallel edges. Figure 3.2 shows a graph G associated with the order picker routing problem in Figure 3.1 where $r = 5$, $m_u = 9$, and $m_l = 8$. Associated with every edge is a weight that corresponds to the distance between the endpoints of that edge.

An order picking tour in the warehouse then corresponds to a tour, i.e. a directed path, on the graph G, and vice versa. Thus, the problem of finding the shortest order picking tour is identical to the problem of finding an order picking tour on the graph G, which will be solved in the next section.

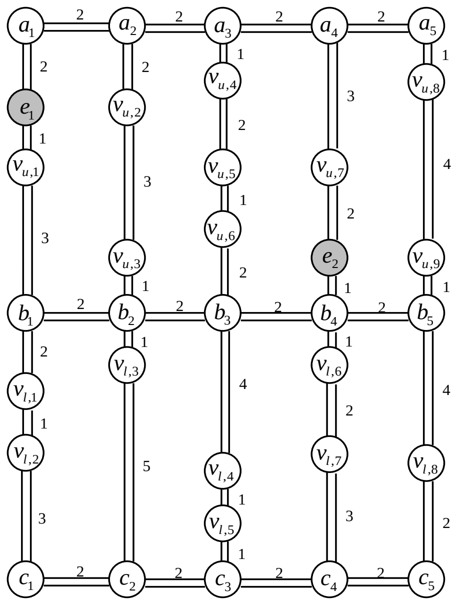


Figure 3.2 Graph representation G, where $r = 5$, $m_u = 9$, and $m_l = 8$.

3.3 Optimal order picker routing with arbitrary starting and ending points

This section presents a solution procedure for the order picker routing problem with arbitrary starting and ending points in a conventional warehouse with two blocks, using the graph representation introduced in the previous section. Löffler et al. (2018) proposed an exact order picker routing algorithm for arbitrary starting and ending points for the case of a conventional warehouse with a single block. We extend their work to a conventional warehouse with two blocks using the solution procedures proposed in the works of RR and Roodbergen and de Koster (2001a).

3.3.1 Outline of the solution procedure

The objective is to find the shortest order picking tour on G. First, we present a procedure for constructing the minimum-length tour subgraph from a graph representation of G. Afterwards, we construct the optimal order picking tour from a minimum-length tour subgraph.

3.3.2 Constructing the minimum-length tour subgraph

To find the minimum-length tour subgraph of G, we consider a certain sequence of increasing subgraphs of G from the left- to the right-most picking aisles that contain items to be picked. In each subgraph along the sequence, partial tour subgraphs (PTSs) and their equivalence classes are considered. In each equivalence class, a PTS with minimum length is selected as a candidate for a PTS of the minimum-length tour subgraph. The idea is to build tour subgraphs by iteratively adding edges from left to right and from the lower block to the upper block. Dynamic programming then ensures that we obtain the tour subgraph of G with minimum length. The solution procedure first constructs PTSs, establishes their equivalence classes, and then uses a dynamic programming procedure to find the minimum-length order picking tour. The different steps of the solution procedure are described in the following.

3.3.2.1 Constructing a partial tour subgraph (PTS)

We first extend the definition of a tour subgraph in RR to the case where the starting and ending points are arbitrarily given.

Definition 3.1 Let e_1 and e_2 be the given starting and ending points of a tour. A subgraph T of G is a tour subgraph if there is an order picking tour that starts in e_1, passes through the vertices $v_{u,i}$ and $v_{l,i}$, and ends in e_2, where each edge in T is traversed exactly once.

We recall that, by definition, an order picking tour must visit every item location, hence the degree of the vertices $v_{u,i}$ and $v_{l,i}$ in T are positive. The starting and ending points e_1 and e_2 can also be interchanged. The interchange will result in the reversed direction of the tour without changing the tour subgraph.

The necessary and sufficient condition for a subgraph T of G to be a tour subgraph is given in Theorem 3.1.

Theorem 3.1 A connected subgraph of an undirected graph is a tour subgraph with starting and ending points e_1 and e_2 if and only if the degrees of e_1 and e_2 in the subgraph are odd, the degrees of the vertices $v_{u,i}$ and $v_{l,i}$ are positive even, and the degrees of the other vertices are even.

Proof. Assume that a connected subgraph T of G is a tour subgraph with starting and ending points e_1 and e_2. By Definition 3.1, there is a tour that starts in e_1, passes through the vertices $v_{u,i}$ and $v_{l,i}$, and ends in e_2, where each edge in T is traversed exactly once. For each vertex $v_{u,i}$, the number of edges adjacent to $v_{u,i}$ that the tour uses to visit $v_{u,i}$ is the same as the number of edges used to leave the vertex. Hence, the degree of $v_{u,i}$ must be even. The same argument implies that the degree of $v_{l,i}$ is even as well. On the other hand, the tour T uses one more edge to leave the vertex e_1, so the degree of the vertex is odd. Similarly, the degree of e_2 is also odd.

Conversely, the degrees of e_1 and e_2 in the subgraph are odd, while the degrees of the other vertices are even. From the Eulerian path theorem, there exists a tour that starts at the vertex e_1, ends at the vertex e_2, and traverses every edge exactly once. Since the degrees of $v_{u,i}$ and $v_{l,i}$ are positive, the tour is an order picking tour. Hence, the subgraph is a tour subgraph.□

By Definition 3.1 and Theorem 3.1, we propose the following characterization of an order picking tour subgraph.

Theorem 3.2 A subgraph $T \subset G$ is an order picking tour subgraph if and only if all the following conditions hold:

1. The vertices $v_{u,i}$ for $i = 1,2,\dots,m_u$, $v_{l,i}$ for $i = 1,2,\dots,m_l$, e_1, and e_2 all have positive degrees in T.
2. The vertices e_1 and e_2 both have odd degrees.
3. Excluding vertices with zero degree, T is connected.
4. Every vertex in T, except e_1 and e_2, has even degree or zero degree.

Proof. Assume that T is an order picking tour subgraph of G. By Definition 3.1, there exists a tour that starts at the vertex e_1, passes through the vertices $v_{u,i}$ and $v_{l,i}$, and ends at the vertex e_2. Hence, the first and the third conditions hold. By Theorem 3.1, the degrees of e_1 and e_2 in T are odd, while the degrees of all other vertices are even. Hence, the second and the fourth conditions hold.

Conversely, the third condition implies that T is connected. The other three conditions then conclude that T is a tour subgraph with starting and ending points e_1 and e_2 that contains all the vertices $v_{u,i}$ and $v_{l,i}$. Hence, T is an order picking tour subgraph.□

The following theorem is useful in constructing a minimum-length order picking tour subgraph.

Theorem 3.3 A minimum-length order picking tour subgraph contains no more than two edges between any pair of vertices.

Proof. By contradiction, assume that there are more than two edges between a pair of vertices in a minimum-length tour subgraph T. We can see that deleting two edges between the two vertices from T will still result in a tour subgraph, but with a shorter length. Hence, the former tour subgraph T cannot be of minimum length. This is a contradiction.□

Definition 3.2 (cf. Ratliff and Rosenthal, 1983). Let L be the subgraph of G, a subgraph T of L is an L PTS if there exists a subgraph C of $G - L$ such that $T \cup C$ is a tour subgraph of G. The subgraph C is called a completion of T.

In our algorithm, we only consider three types of the subgraphs L, which are described in the following definition.

Definition 3.3 (cf. Roodbergen and de Koster, 2001a) Let L_j^- be a subgraph of G containing the vertices a_j, b_j, c_j, and all vertices and edges to the left of them. Let l_j be a subgraph of G containing the vertices b_j, c_j, and all vertices and edges between them. We define $L_j^{+l} = L_j^- \cup l_j$. Similarly, let u_j be a subgraph of G consisting of the vertices a_j and b_j together with all vertices and edges between them and define $L_j^{+u} = L_j^{+l} \cup u_j$. From this point forward, L_j will be used when a result holds if we let $L_j = L_j^-$, $L_j = L_j^{+l}$, or $L_j = L_j^{+u}$.

To find the minimum-length tour subgraph of G, we consider a sequence of increasing subgraphs of G from aisle $j = 1$ to aisle $j = r$, where L_j PTSs for each aisle j are considered. The following theorem extends the theorem in Roodbergen and de Koster (2001a) and gives the necessary and sufficient conditions for a subgraph of G to be an L_j PTS.

Theorem 3.4 A subgraph $T_j \subset L_j$ is an L_j PTS if and only if

1. The degrees of all $v_{u,i}, v_{l,i} \in L_j$ are positive in T_j.
2. If L_j contains the vertex e_i ($i = 1$ or 2), the degree of e_i is odd.
3. Every vertex in T_j, except possibly for a_j, b_j, c_j, and e_i, has even or zero degree.
4. Excluding vertices with zero degree, T_j has either
   - no connected component,
   - a single connected component containing at least one of a_j, b_j, and c_j,
   - two connected components, with each component containing at least one of a_j, b_j, and c_j,
   - three connected components with a_j, b_j, and c_j each in a different component.

Proof. To prove necessity, we extend the proof of RR. We assume that T_j is an L_j PTS. By Definition 3.2, there exists a subgraph C_j of $G - L_j$ such that $T = T_j \cup C_j$ is a tour subgraph. Since the first three conditions hold in T, they consequently hold in T_j as well. We are left to show that the fourth condition holds. Note that, except possibly for a_j, b_j, and c_j, there are no vertices in C_j incident to the vertices in T_j. Since a tour subgraph T is connected, each connected component of T_j must contain at least one of a_j, b_j, and c_j. Hence, the fourth condition holds.

To prove sufficiency, we extend the proof of RR and Löffler et al. (2018). We assume that all conditions hold. In order to show that a subgraph T_j is an L_j PTS, we have to find a subgraph C_j of $G - L_j$ such that $T_j \cup C_j$ is a tour subgraph. From Theorem 3.2, C_j must have the following properties:

1. It contains all vertices $v_{u,i}$, $v_{l,i} \in G - L_j$, and their degrees in C_j are positive even.
2. It contains all vertices $e_i \in G - L_j$, and their degrees in C_j are odd.
3. Excluding vertices with zero degree, $T_j \cup C_j$ is connected.

We note that T_j can contain both e_1 and e_2, either e_1 or e_2, or neither e_1 nor e_2. In each case, a completion C_j can be easily constructed as shown in Figure 3.3 by taking into account the presence of the vertices e_i in T_j. Configuration 1 represents a completion C_j when T_j contains both of e_1 and e_2, while configuration 2 is a completion C_j for the case when T_j contains either e_1 or e_2. Both configurations 3.1 and 3.2 represent completions C_j when T_j contains neither e_1 nor e_2; configuration 3.1 is applied when the degree parities of a_j, b_j, and c_j are all even in L_j, whereas configuration 3.2 is used when the degree parities of a_j and b_j are all odd and the degree parity of c_j is even (The other cases can be done similarly). Note that parallel dashed edges in the upper and lower blocks of aisle j are included in C_j for the case $L_j = L_j^-$. Conversely, they are all excluded from C_j when $L_j = L_j^{+u}$. For $L_j = L_j^{+l}$, parallel dashed edges in the upper block are included in C_j, while parallel dashed edges in the lower block are excluded.□

3.3.2.2 Equivalence class of a partial tour subgraph (PTS)

As mentioned in Section 3.3.2, to construct the minimum-length tour subgraph of G, partial tour subgraphs and their equivalence classes have to be considered simultaneously. Suppose that there are two L_j PTSs, T_j^1 and T_j^2. The equivalence of these two PTSs is specified in Definition 3.4.

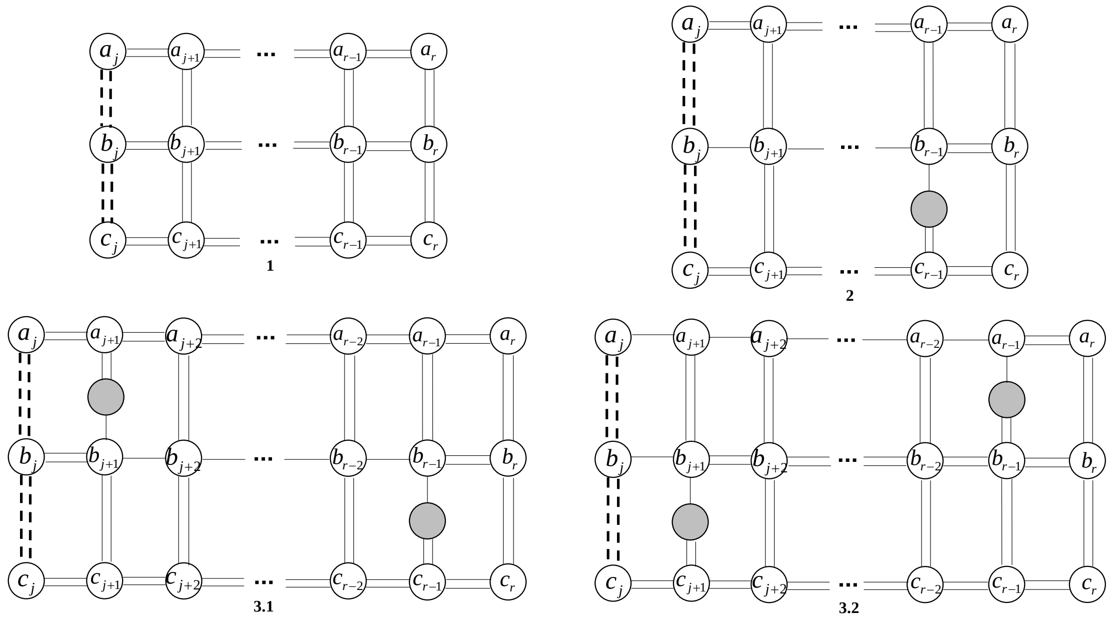


Figure 3.3 Completions C_j of T_j such that $T_j \cup C_j$ is a tour subgraph.

Definition 3.4 (cf. Ratliff and Rosenthal, 1983) Two L_j PTSs T_j^1 and T_j^2 are equivalent if for any $C_j \subset G - L_j$ such that $T_j^1 \cup C_j$ is a tour subgraph, $T_j^2 \cup C_j$ is also a tour subgraph, and vice versa.

The conditions for two L_j PTSs to be equivalent are described in Theorem 3.5.

Theorem 3.5 (cf. Roodbergen and de Koster, 2001a) Two L_j PTSs are equivalent if and only if

1. a_j, b_j, and c_j each have the same degree parity (i.e., even (including 0) or odd) in both PTSs.
2. Excluding vertices with zero degree, both PTSs have either
   - no connected component,
   - a single connected component containing at least one of a_j, b_j, and c_j,
   - two connected components, with at least one of a_j, b_j, and c_j in each component,
   - three connected components, with a_j, b_j, and c_j each in a different component.
3. The distribution of a_j, b_j, and c_j over the various components is the same for both PTSs.

Proof. We extend the proof of RR. Assume that T_j^1 and T_j^2 are equivalent L_j PTSs. By Definition 3.2, consider any subgraph C_j of $G - L_j$ such that $T_j^1 \cup C_j$ is a tour subgraph, then $T_j^2 \cup C_j$ is also a tour subgraph. Since $T_j^1 \cup C_j$ is a tour subgraph, the degrees of a_j, b_j, and c_j in $T_j^1 \cup C_j$ must be even. Therefore, the degree parity of these vertices in T_j^1 and C_j are the same. The same argument holds for T_j^2. Hence, a_j, b_j, and c_j have the same degree parity in T_j^1 and T_j^2.

If T_j^1 and T_j^2 have different numbers of connected components or different distributions of a_j, b_j, and c_j over the components, we can easily find a subgraph C_j that is a completion of either T_j^1 or T_j^2, but not of the respective other. Hence, all three conditions hold.

Conversely, let T_j^1 and T_j^2 satisfy the three conditions. Let C_j be a subgraph of $G - L_j$ such that $T_j^1 \cup C_j$ is a tour subgraph. Since T_j^1, T_j^2, and C_j are PTSs, the degrees of e_1 and e_2 are odd, while the degrees of the other vertices except possibly for a_j, b_j, and c_j are even. Since a_j, b_j, and c_j have the same degree parity in T_j^1 and C_j, they also have the same degree parity in T_j^2. Thus, the degrees of a_j, b_j, and c_j in $T_j^2 \cup C_j$ are even.

We are left to show that $T_j^2 \cup C_j$ is connected. Observe that shrinking a connected component of a graph does not change its degree parity and the connectivity of the graph. If we shrink the connected components of T_j^1 and T_j^2 to single vertices, the resulting graphs still have the same degree parity and connectivity. Therefore, since $T_j^1 \cup C_j$ is connected, $T_j^2 \cup C_j$ is also connected. By Theorem 3.2, $T_j^2 \cup C_j$ is a tour subgraph. Hence, T_j^1 and T_j^2 are equivalent.□

From Theorem 3.5, an equivalence class of L_j PTSs can be represented by the degree parity of $a_j(\deg(a_j))$, degree parity of $b_j(\deg(b_j))$, degree parity of $c_j(\deg(c_j))$, and their connectivity in the PTS. The degree parities of a_j, b_j, and c_j can be even (E) (including zero (0)) or uneven (U). In order to decrease the number of equivalence classes in the solution procedure, we do not separate the degree parity between zero and even, but instead use the pattern of connectivity to decide on the degree parities of a_j, b_j, and c_j in case their degree parities are even.

The connectivity encodes the number of components as well as the distribution of a_j, b_j, and c_j over those components. From Theorem 3.5, it follows that there are fifteen possible types of connectivity, which are illustrated in Table 3.1. The connectivity of a PTS that does not have any connected component is indicated by the symbol (-). A single connected component containing at least one of a_j, b_j, and c_j is represented by a, b, c, ab, ac, bc, and abc. Two connected components with at least one of a_j, b_j, and c_j in each component are represented by $a - b$, $a - c$, $b - c$, $a - bc$, $b - ac$, and $c - ab$. Three connected components with a_j, b_j, and c_j each in a different component is represented by $a - b - c$.

For instance, the equivalence class $EUU(a - bc)$ indicates that the degree parities of a_j, b_j, and c_j are even, uneven, and uneven, respectively, and there are two connected components with a_j in one component, while b_j and c_j are in the other. Since $\deg(a_j)$, $\deg(b_j)$, and deg(c_j) can be either even or uneven and the number of possible types of connectivity is fifteen, there are $2 \times 2 \times 2 \times 15 = 120$ equivalence classes that are relevant for the solution procedure. However, these 120 equivalence classes contain some classes that lead to infeasible solutions. As an example, $EEU(a-b)$ has E, E, and U in a_j, b_j, and c_j, respectively, and consists of two connected components with a_j in one component and b_j in the other. It is impossible that the degree parity of c_j is uneven (odd) since there is no connected component containing c_j. After deleting all equivalence classes that lead to infeasible solutions, we obtain 70 possible equivalence classes. From Theorem 3.4, all possible equivalence classes of an L_j PTS can be categorized as described in Theorems 3.6, 3.7, and 3.8. From Theorem 3.2, since a tour subgraph must be connected and possess even degree, the shortest length from $EEE(a)$, $EEE(b)$, $EEE(c)$, $EEE(ab)$, $EEE(ac)$, $EEE(bc)$, and $EEE(abc)$ of L_r^{+u} in the last aisle will be the minimum-length tour subgraph.

Theorem 3.6 In case L_j contains neither e_1 nor e_2, the sum of degree parities of a_j, b_j, and c_j of an L_j PTS is even, and the possible equivalence classes are: $EEE(-)$, $EEE(a)$, $EEE(b)$, $EEE(c)$, $EEE(ab)$, $EEE(ac)$, $EEE(bc)$, $EEE(abc)$, $EEE(a-b)$, $EEE(a-c)$, $EEE(b-c)$, $EEE(a-bc)$, $EEE(b-ac)$, $EEE(c-ab)$, $EEE(a-b-c)$, $EUU(bc)$, $EUU(abc)$, $EUU(a-bc)$, $UEU(ac)$, $UEU(abc)$, $UEU(b-ac)$, $UUE(ab)$, $UUE(abc)$, $UUE(c-ab)$.

Theorem 3.7 In case L_j contains exactly one of e_1 and e_2, the sum of degree parities of a_j, b_j, and c_j of an L_j PTS is odd, and the possible equivalence classes are: $EEU(c)$, $EEU(ac)$, $EEU(bc)$, $EEU(abc)$, $EEU(a-c)$, $EEU(b-c)$, $EEU(a-bc)$, $EEU(b-ac)$, $EEU(c-ab)$, $EEU(a-b-c)$, $EUE(b)$, $EUE(ab)$, $EUE(bc)$, $EUE(abc)$, $EUE(a-b)$, $EUE(b-c)$, $EUE(a-bc)$, $EUE(b-ac)$, $EUE(c-ab)$, $EUE(a-b-c)$, $UEE(a)$, $UEE(ab)$, $UEE(ac)$, $UEE(abc)$, $UEE(a-b)$, $UEE(a-c)$, $UEE(a-bc)$, $UEE(b-ac)$, $UEE(c-ab)$, $UEE(a-b-c)$, $UUU(abc)$, $UUU(a-bc)$, $UUU(b-ac)$, $UUU(c-ab)$.

Theorem 3.8 In case L_j contains both e_1 and e_2, the sum of degree parities of a_j, b_j, and c_j of an L_j PTS is even, and the possible equivalence classes are: $EEE(-)$, $EEE(a)$, $EEE(b)$, $EEE(c)$, $EEE(ab)$, $EEE(ac)$, $EEE(bc)$, $EEE(abc)$, $EEE(a-b)$, $EEE(a-c)$, $EEE(b-c)$, $EEE(a-bc)$, $EEE(b-ac)$, $EEE(c-ab)$, $EEE(a-b-c)$, $EUU(bc)$, $EUU(abc)$, $EUU(b-c)$, $EUU(a-bc)$, $EUU(b-ac)$, $EUU(c-ab)$, $EUU(a-b-c)$, $UEU(ac)$, $UEU(abc)$, $UEU(a-c)$, $UEU(a-bc)$, $UEU(b-ac)$, $UEU(c-ab)$, $UEU(a-b-c)$, $UUE(ab)$, $UUE(abc)$, $UUE(a-b)$, $UUE(a-bc)$, $UUE(b-ac)$, $UUE(c-ab)$, $UUE(a-b-c)$.

3.3.2.3 Dynamic programming procedure

In the proposed dynamic programming procedure, we define the states as the equivalence classes of PTSs as well as the three different transitions between states that consist of adding vertices and edges to L_j PTSs, L_j^- to L_j^{+l}, L_j^{+l} to L_j^{+u}, and L_j^{+u} to L_{j+1}^- (Roodbergen and de Koster, 2001a). Since a minimum-length tour subgraph contains no more than two edges between any two adjacent vertices (see Theorem 3.3), we consider only the vertical and horizontal components with a single edge and/or double edges between any pair of vertices in each transition. The three possible types of transitions are described in the following.

1. *Transition from* L_j^- to L_j^{+l}

This transition transforms an L_j^- PTS to an L_j^{+l} PTS by adding a vertical component for traversing aisle j of the lower block to the L_j^- PTS. The recursive formula (1) provides a connection between the objective value of the previous and the current state. The value $\omega(z, e)$ denotes the objective value of state (z, e), i.e., the shortest length z PTS of equivalence class e, where e belongs to one of the equivalence classes (*Eq*) described in Theorems 3.6 to 3.8. The value $\omega(v)$ represents the added distance from the vertical transition corresponding to the vertical components in Figure 3.4, where v belongs to one of the vertical components (*Ver*) in Figure 3.4.

$$\omega\left(L_j^{+l}, e\right) = \min_{v \in \{Ver\}, e \in \{Eq\}} \{\omega\left(L_j^-, e\right) + \omega(v)\} \qquad (1)$$

Figure 3.4 shows all seventeen possible vertical components between the vertices b_j and c_j that are candidates for the optimal solution. They can be categorized into three main groups. Configurations 1-6 represent the vertical components for the case where aisle j of the lower block contains neither the starting point e_1 nor the ending point e_2. If aisle j contains either e_1 or e_2, configurations 7-12 are applied. Configurations 13-17 are finally used when aisle j contains both e_1 and e_2. Configurations 17a and 17b are shown for clarity, and both will be referred to as configuration 17 in the following. In configuration 4, the largest gap between two adjacent items is not traversed. For configurations 11, 12, and 17, the largest gap between any two adjacent items on either side of e_1 or e_2 is not traversed. In configuration 13, the largest gap between e_1 and e_2 is not traversed. Configuration 6 can only be selected if aisle j in the lower block is empty. The connectivity of the L_j^{+l} PTS equivalence classes as well as deg(a_j), deg(b_j), deg(c_j) that result from adding vertical components illustrated in Figure 3.4 to the L_j^- PTS equivalence classes are given in Table 3.1. For instance, if deg(a_j), deg(b_j), deg(c_j), and connectivity of an L_j^- PTS are U, E, E, and a, respectively, the L_j^- PTS belongs to the $UEE(a)$ equivalence class. If it is expanded with vertical component configuration 3, an L_j^{+l} PTS that results from this transition belongs to the class $UEE(a-c)$. An example for the case where $j = 3$ is illustrated in Figure 3.5.

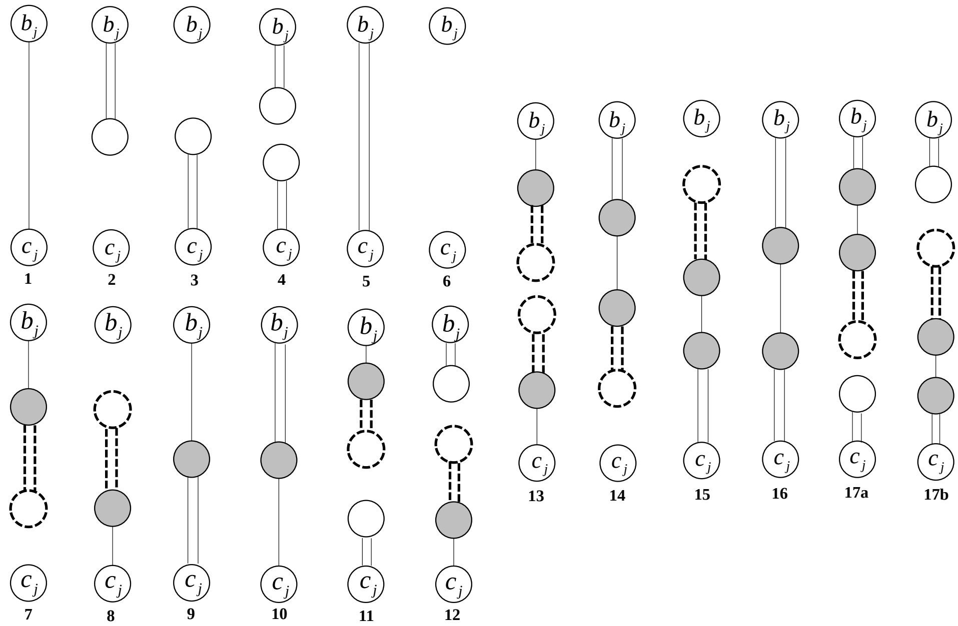


Figure 3.4 Vertical components for aisle j in the lower block. Dashed parts are optional.

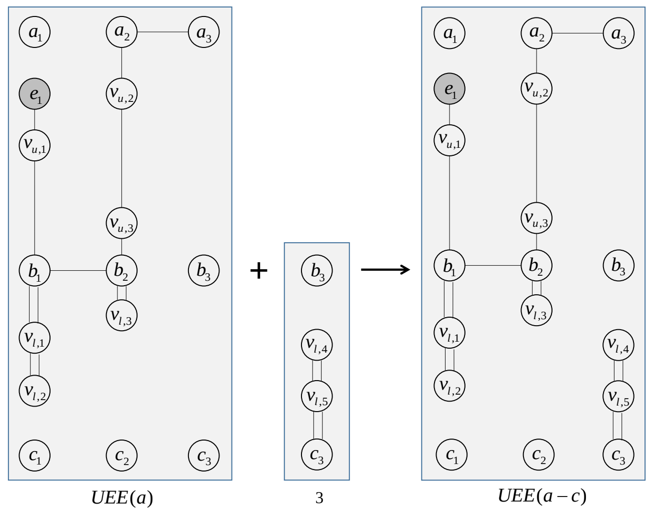


Figure 3.5 Adding vertical component configuration 3 to an L_3^- PTS of type $UEE(a)$ results in an L_3^{+l} PTS of type $UEE(a-c)$.

2. *Transition from L_j^{+l} to L_j^{+u}*

The transition is similar to the previous transition, but a vertical component for traversing between a_j and b_j is added to an L_j^{+l} PTS instead. Thus, the same seventeen vertical components described earlier can be used. The connectivity of the L_j^{+u} PTS equivalence classes as well as deg(a_j), deg(b_j), and deg(c_j) are given in Table 3.2. The following Equation (2) is the recursive formula for this transition.

$$\omega\left(L_j^{+u}, e\right) = \min_{v \in \{Ver\}, e \in \{Eq\}} \{\omega\left(L_j^{+l}, e\right) + \omega(v)\} \quad (2)$$

3. *Transition from L_j^{+u} to L_{j+1}^{-}*

This transition transforms an L_j^{+u} PTS to an L_{j+1}^{-} PTS by connecting aisle j to aisle $j+1$ by a horizontal component between a_j and a_{j+1}, b_j and b_{j+1}, and c_j and c_{j+1}. If an L_j^{+u} PTS contains neither e_1 nor e_2 or both of them, the horizontal components 1-14 in Figure 3.6 are applied. If the L_j^{+u} PTS contains either e_1 or e_2, horizontal components 1-13 in Figure 3.7 are used in this transition. All horizontal components shown in Figure 3.6 and Figure 3.7 can be grouped by their connectivity into eight main types: -, a, b, c, $a-b$, $a-c$, $b-c$, and $a-b-c$. The symbol (-) represents the empty horizontal component shown as component 14 in Figure 3.6. The recursive formula (3) provides a connection between the objective value of the previous states and the current state, where $\omega(h)$ represents the added distance from the horizontal transition corresponding to the horizontal components in Figure 3.6 and Figure 3.7, and where h belongs to one of the horizontal components (Hor) in Figure 3.6 and Figure 3.7.

$$\omega\left(L_{j+1}^{-}, e'\right) = \min_{h \in \{Hor\}, e \in \{Eq\}} \{\omega\left(L_j^{+u}, e\right) + \omega(h)\} \quad (3)$$

Notice, for example, that both configuration 4 in Figure 3.6 and configuration 1 in Figure 3.7 have the connectivity pattern a, but the former is used only when neither e_1 nor e_2 or both of them are contained in L_j^{+u} PTSs, while the latter is used when either e_1 or e_2 are contained in L_j^{+u} PTSs. Adding a horizontal component from Figure 3.6 or Figure 3.7 to an L_j^{+u} PTS creates an L_{j+1}^{-} PTS. Table 3.3 shows deg(a_j) of the resulting L_{j+1}^{-} PTS equivalence classes. We note that not every horizontal component can be used due to Theorem 3.4 (condition 3). For example, if deg(a_j) in an L_j^{+u} PTS is even, deg(a_j) of a horizontal component must be even, and the resulting L_{j+1}^{-} PTS equivalence class will be of even degree as well. The tables for the degree of b_j and c_j can be easily derived and are identical to Table 3.3. Similarly, Table 3.4 shows the connectivity of the L_{j+1}^{-} PTS that results from adding a horizontal component of different patterns of connectivity. For example, if an L_j^{+u} equivalence class of connectivity type $a-bc$ is extended by a horizontal component of type $a-b$, the connectivity of the resulting L_{j+1}^{-} PTS will be $a-b$.

In terms of the run-time complexity of the proposed algorithm, it is linear in the number of requested items m and picking aisles r, i.e. $O(m+r)$.

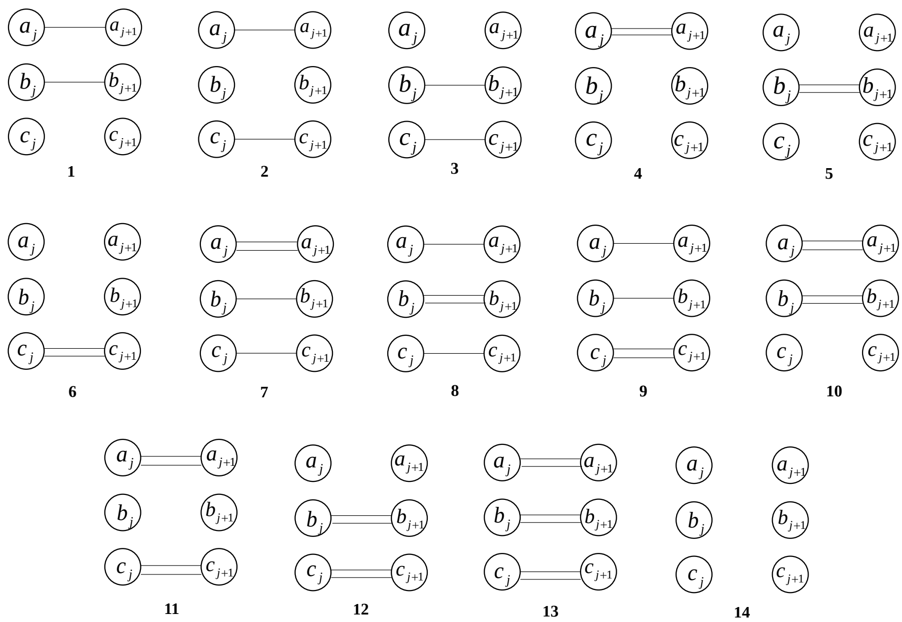


Figure 3.6 Horizontal components for travelling from aisle j to aisle $j+1$ when an L_j^{+u} PTS contains neither e_1 nor e_2 or both of them.

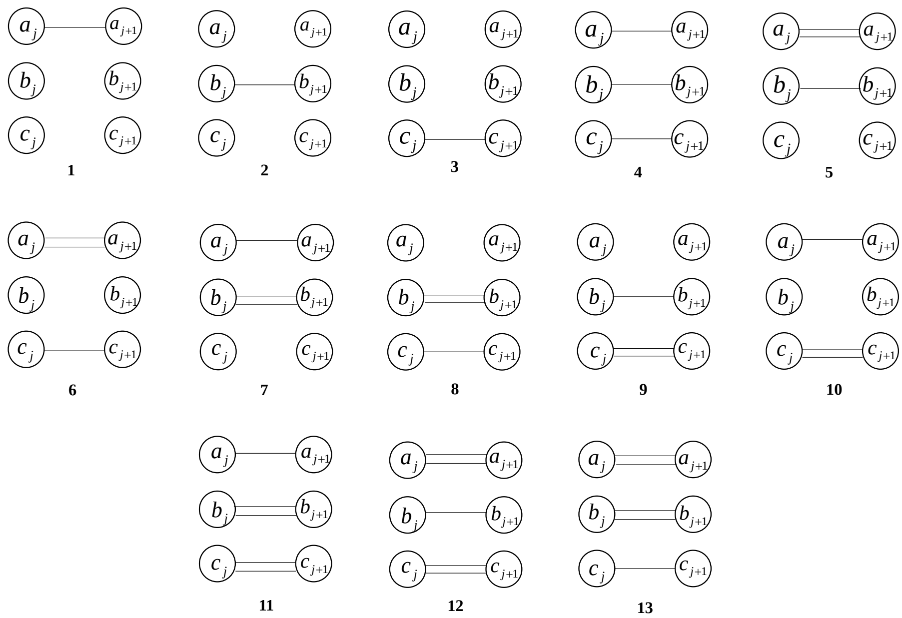


Figure 3.7 Horizontal components for travelling from aisle j to aisle $j+1$ when an L_j^{+u} PTS contains either e_1 or e_2.

Table 3.1 Degree parities of a_j, b_j, c_j, and connectivity of L_j^{+l} PTS equivalence class.

L_j^- PTS Equivalence class		Vertical components that can be used for traversing aisle j of the lower block (between b_j and c_j)												
		when aisle j of the lower block contains neither e_1 nor e_2 or both of them							when aisle j of the lower block contains either e_1 or e_2					
		1	2,14	3,15	4,17	5,16	6	13	7	8	9	10	11	12
deg(a_j)	E	E	E	E	E	E	E	E	E	E	E	E	E	E
	U	U	U	U	U	U	U	U	U	U	U	U	U	U
deg(b_j)	E	U	E	E	E	E	E	U	U	E	U	E	U	E
	U	E	U	U	U	U	U	E	E	U	E	U	E	U
deg(c_j)	E	U	E	E	E	E	E	U	E	U	E	U	E	U
	U	E	U	U	U	U	U	E	U	E	U	E	U	E
Connectivity	-	bc	b	c	b-c	bc	-	b-c	b	c	bc	bc	b-c	b-c
	a	a-bc	a-b	a-c	a-b-c	a-bc	a	a-b-c	a-b	a-c	a-bc	a-bc	a-b-c	a-b-c
	b	bc	b	b-c	b-c	bc	b	b-c	b	b-c	bc	bc	b-c	b-c
	c	bc	b-c	c	b-c	bc	c	b-c	b-c	c	bc	bc	b-c	b-c
	ab	abc	ab	c-ab	c-ab	abc	ab	c-ab	ab	c-ab	abc	abc	c-ab	c-ab
	ac	abc	b-ac	ac	b-ac	abc	ac	b-ac	b-ac	ac	abc	abc	b-ac	b-ac
	bc	bc	bc	bc	bc	bc	bc	bc	bc	bc	bc	bc	bc	bc
	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc	abc
	a-b	a-bc	a-b	a-b-c	a-b-c	a-bc	a-b	a-b-c	a-b	a-b-c	a-bc	a-bc	a-b-c	a-b-c
	a-c	a-bc	a-b-c	a-c	a-b-c	a-bc	a-c	a-b-c	a-b-c	a-c	a-bc	a-bc	a-b-c	a-b-c
	b-c	bc	b-c	b-c	b-c	bc	b-c	b-c	b-c	b-c	bc	bc	b-c	b-c
	a-bc	a-bc	a-bc	a-bc	a-bc	a-bc	a-bc	a-bc	a-bc	a-bc	a-bc	a-bc	a-bc	a-bc
	b-ac	abc	b-ac	b-ac	b-ac	abc	b-ac	b-ac	b-ac	b-ac	abc	abc	b-ac	b-ac
	c-ab	abc	c-ab	c-ab	c-ab	abc	c-ab	c-ab	c-ab	c-ab	abc	abc	c-ab	c-ab
	a-b-c	a-bc	a-b-c	a-b-c	a-b-c	a-bc	a-b-c	a-b-c	a-b-c	a-b-c	a-bc	a-bc	a-b-c	a-b-c

Table 3.2 Degree parities of a_j, b_j, c_j, and connectivity of L_j^{+u} PTS equivalence class.

L_j^{+l} PTS Equivalence class		Vertical components that can be used for traversing aisle j of the upper block (between a_j and b_j)												
		when aisle j of the upper block contains neither e_1 nor e_2 or both of them							when aisle j of the upper block contains either e_1 or e_2					
		1	2,14	3,15	4,17	5,16	6	13	7	8	9	10	11	12
deg(a_j)	*E*	*U*	*E*	*E*	*E*	*E*	*E*	*U*	*U*	*E*	*U*	*E*	*U*	*E*
	U	*E*	*U*	*U*	*U*	*U*	*U*	*E*	*E*	*U*	*E*	*U*	*E*	*U*
deg(b_j)	*E*	*U*	*E*	*E*	*E*	*E*	*E*	*U*	*E*	*U*	*E*	*U*	*E*	*U*
	U	*E*	*U*	*U*	*U*	*U*	*U*	*E*	*U*	*E*	*U*	*E*	*U*	*E*
deg(c_j)	*E*	*E*	*E*	*E*	*E*	*E*	*E*	*E*	*E*	*E*	*E*	*E*	*E*	*E*
	U	*U*	*U*	*U*	*U*	*U*	*U*	*U*	*U*	*U*	*U*	*U*	*U*	*U*
connectivity	-	*ab*	*a*	*b*	*a-b*	*ab*	-	*a-b*	*a*	*b*	*ab*	*ab*	*a-b*	*a-b*
	a	*ab*	*a*	*a-b*	*a-b*	*ab*	*a*	*a-b*	*a*	*a-b*	*ab*	*ab*	*a-b*	*a-b*
	b	*ab*	*a-b*	*b*	*a-b*	*ab*	*b*	*a-b*	*a-b*	*b*	*ab*	*ab*	*a-b*	*a-b*
	c	*c-ab*	*a-c*	*b-c*	*a-b-c*	*c-ab*	*c*	*a-b-c*	*a-c*	*b-c*	*c-ab*	*c-ab*	*a-b-c*	*a-b-c*
	ab	*ab*	*ab*	*ab*	*ab*	*ab*	*ab*	*ab*	*ab*	*ab*	*ab*	*ab*	*ab*	*ab*
	ac	*abc*	*ac*	*b-ac*	*b-ac*	*abc*	*ac*	*b-ac*	*ac*	*b-ac*	*abc*	*abc*	*b-ac*	*b-ac*
	bc	*abc*	*a-bc*	*bc*	*a-bc*	*abc*	*bc*	*a-bc*	*a-bc*	*bc*	*abc*	*abc*	*a-bc*	*a-bc*
	abc	*abc*	*abc*	*abc*	*abc*	*abc*	*abc*	*abc*	*abc*	*abc*	*abc*	*abc*	*abc*	*abc*
	a-b	*ab*	*a-b*	*a-b*	*a-b*	*ab*	*a-b*	*a-b*	*a-b*	*a-b*	*ab*	*ab*	*a-b*	*a-b*
	a-c	*c-ab*	*a-c*	*a-b-c*	*a-b-c*	*c-ab*	*a-c*	*a-b-c*	*a-c*	*a-b-c*	*c-ab*	*c-ab*	*a-b-c*	*a-b-c*
	b-c	*c-ab*	*a-b-c*	*b-c*	*a-b-c*	*c-ab*	*b-c*	*a-b-c*	*a-b-c*	*b-c*	*c-ab*	*c-ab*	*a-b-c*	*a-b-c*
	a-bc	*abc*	*a-bc*	*a-bc*	*a-bc*	*abc*	*a-bc*	*a-bc*	*a-bc*	*a-bc*	*abc*	*abc*	*a-bc*	*a-bc*
	b-ac	*abc*	*b-ac*	*b-ac*	*b-ac*	*abc*	*b-ac*	*b-ac*	*b-ac*	*b-ac*	*abc*	*abc*	*b-ac*	*b-ac*
	c-ab	*c-ab*	*c-ab*	*c-ab*	*c-ab*	*c-ab*	*c-ab*	*c-ab*	*c-ab*	*c-ab*	*c-ab*	*c-ab*	*c-ab*	*c-ab*
	a-b-c	*c-ab*	*a-b-c*	*a-b-c*	*a-b-c*	*c-ab*	*a-b-c*	*a-b-c*	*a-b-c*	*a-b-c*	*c-ab*	*c-ab*	*a-b-c*	*a-b-c*

Table 3.3 deg(a_j), deg(b_j), and deg(c_j) of the resulting L_{j+1}^{-} PTS equivalence class.

deg(a_j), deg(b_j), and deg(c_j) in L_j^{+u}	deg(a_j), deg(b_j), and deg(c_j) of horizontal component	
	E	U
E	E	
U		U

Table 3.4 Connectivity of L_{j+1}^{-} equivalence class.

Connectivity of L_j^{+u} PTS	Connectivity of the horizontal component added to the L_j^{+u} PTS							
	-	*a*	*b*	*c*	*a-b*	*a-c*	*b-c*	*a-b-c*
-	-	*	*	*	*	*	*	*
a	#	*a*	&	&	*	*	&	*
b	#	&	*b*	&	*	&	*	*
c	#	&	&	*c*	&	*	*	*
ab	#	*a*	*b*	&	*ab*	*	*	*
ac	#	*a*	&	*c*	*	*ac*	*	*
bc	#	&	*b*	*c*	*	*	*bc*	*
abc	#	*a*	*b*	*c*	*ab*	*ac*	*bc*	*abc*
a-b	&	&	&	&	*a-b*	&	&	*
a-c	&	&	&	&	&	*a-c*	&	*
b-c	&	&	&	&	&	&	*b-c*	*
a-bc	&	&	&	&	*a-b*	*a-c*	&	*a-bc*
b-ac	&	&	&	&	*a-b*	&	*b-c*	*b-ac*
c-ab	&	&	&	&	&	*a-c*	*b-c*	*c-ab*
a-b-c	&	&	&	&	&	&	&	*a-b-c*

* This transition would never give the optimal solution.
\# This transition can occur only if there are no items to be picked in $G - L_j^{+u}$.
& This transition would contradict the condition 4 in Theorem 3.4.

3.3.3 Tour construction algorithm

Section 3.3.2 outlined a procedure for finding the minimum-length tour subgraph that contains the locations of the m required items as well as the starting point e_1 and the ending point e_2. This section presents a procedure to construct a minimum-length order picking tour in the warehouse from the minimum-length tour subgraph of a graph representation G. In the following, we adapt the tour construction procedure presented in RR to our problem. The procedure is described in the following:

Step 1. Begin the order picking tour at the starting point e_1 as the first vertex visited.

Step 2. Let v^* be the vertex currently being visited.

Step 3. If there is a pair of unused parallel edges incident to v^*, choose one of them to move to the next vertex, then go back to *Step 2*. Otherwise, continue to the next step.

Step 4. If there are unused single edges that are not a pair of parallel edges from *Step 3*, incident to v^*, choose one of them to move to the next vertex, then go back to *Step 2*. Otherwise, continue to the next step.

Step 5. If there is a pair of parallel edges incident to v^* including an unused edge, choose it to move to the next vertex, then go back to *Step 2*. Otherwise, continue to the next step.

Step 6. The order picking tour is at the ending point e_2 as the last vertex visited. The minimum-length order picking tour is complete.

3.4 Numerical example

This section illustrates a numerical example to illustrate the solution procedure using the example in **Figure 3.2** Graph representation G, where $r = 5$, $m_u = 9$, and $m_l = 8$. To find the minimum-length tour subgraph of G, we follow the dynamic programming procedure presented in Section 3.3.2.3 on the sequence of iteratively-built PTSs. We start from the unique L_1^- PTS of type $EEE(-)$. Since there are items to be picked in the lower block of the left-most picking aisle, and since this aisle neither contains the starting point e_1 nor the ending point e_2, vertical components 1-5 are selected from **Table 3.1** to construct the L_1^{+l} PTSs and their equivalence classes. The resulting L_1^{+l} PTSs are $EUU(bc)$, $EEE(b)$, $EEE(c)$, $EEE(b-c)$, and $EEE(bc)$ with their minimum lengths 6, 6, 8, 10, and 12, respectively. To create the L_1^{+u} PTSs, we add vertical components 7-12 from **Figure 3.4** to all five L_1^{+l} PTSs that result from the previous transition. The PTSs with their minimum lengths are selected as candidates for the PTSs of the minimum-length tour subgraph. To move from aisle 1 to aisle 2, since the L_1^{+u} PTSs contain e_1, only horizontal components 1-13 in **Figure 3.7** are applicable. For each L_1^{+u} PTS, we then use **Table 3.3** to rule out some invalid horizontal components and obtain L_2^- PTSs using **Table 3.4**. For example, an L_1^{+u} PTS of type $UUU(a-bc)$ can be extended only by horizontal component 4 in **Figure 3.7**. Then, the minimum-length PTSs in L_j^-, L_j^{+l}, and L_j^{+u} for aisles $j = 2, \dots, 5$ can be calculated in the same manner. This solution procedure was coded in Java, and we obtain the minimum-length tour subgraph of length 65 as shown in **Figure 3.8**. To construct a minimum-length order picking tour from this subgraph, the tour construction algorithm proposed in Section 3.3.3 is used.

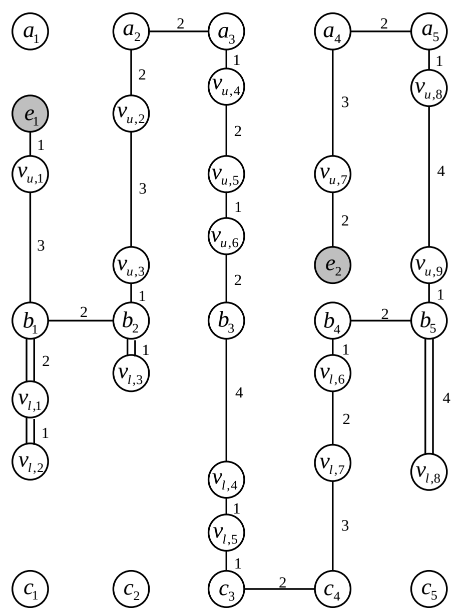


Figure 3.8 The minimum-length tour subgraph of the graph representation G from **Figure 3.2**.

3.5 Computational experiments

In this section, we conduct computational experiments to first study the impact of the middle cross aisle on the performance of the warehouse in case the starting and ending points of a picking tour can be any location in the warehouse. Secondly, we evaluate the performance of a simple routing heuristic as compared to the exact algorithm for the same scenario. The proposed algorithm and the heuristic were implemented in Java, and all instances were run on a computer with Intel Core i5-7200U 2.50 GHz and 8 GB RAM. Since the average computational time for each instance in Sections 3.5.1 and 3.5.2 is less than 1 second, we conclude that the computational time is less of an issue for the routing procedures we propose in this chapter.

3.5.1 Impact of the middle cross aisle on the performance of the warehouse

Roodbergen and de Koster (2001a) conducted numerical experiments to study the impact of the middle cross aisle where a picking tour starts and ends at the depot located at the front of a picking aisle. In order to evaluate the impact of the middle cross aisle on the performance of a warehouse where the starting and ending points of a tour can be any location in the warehouse, we carry out computational experiments to compare the average order picking tour length in the warehouses with and without the middle cross aisle using the exact algorithm proposed in this study. Parameter values varied in the experiments include the size of the warehouse, the layout of the warehouse (with and without the middle cross aisle and the number of aisles), and the size of the pick-list, with some problem sets taken from Roodbergen and de

Koster (2001a). We consider two warehouse sizes with a total aisle length of 70 and 450 meters. The distance between two neighboring storage locations and two adjacent picking aisles are set to 1 and 3 meter(s), respectively. The distance from the respective last picking positions in the parallel aisles to a cross aisle is set to 1 meter.

To study the impact of the middle cross aisle on the performance of the warehouse, we compare the average order picking tour length of both warehouses (with and without the middle cross aisle) on their optimal layouts. This means that we determine the optimal number of aisles for both warehouse types for each pick-list size. To do so, we calculate the average tour length for a fixed pick-list size by varying the number of aisles from 1 to 50. For example, for the optimal number of aisles in the warehouse with a total aisle length of 450 meters and a pick-list size of 30, we consider warehouses with 1 aisle of 450 meters, 2 aisles of 225 meters, …, and 50 aisles of 9 meters. For each warehouse layout, we then randomly generate 1,000 orders with 30 items per pick-list and calculate the average tour length for the warehouse without the middle cross aisle. After that, the middle cross aisle is added to the warehouse and the average tour length is calculated for the same 1,000 orders. Figure 3.9 shows the average tour length in both warehouses (with and without the middle cross aisle) with a total aisle length of 450 meters and a pick-list size of 30 items. We find the optimal number of aisles for both warehouse layouts by choosing the minimum of the tour length curve in Figure 3.9. The optimal number of aisles for the warehouses with and without the middle cross aisle are 25 and 30 with average tour lengths of 293.7 and 358.2 meters, respectively. We apply the same procedure to find the optimal number of aisles and the average tour lengths for all pick-list sizes from 1 to 50.

Figure 3.10 presents the average tour lengths for each pick-list size ranging from 1 to 50 for warehouses with and without the middle cross aisle with total aisle length of 450 meters. The average tour lengths for the warehouse with the middle cross aisle are lower than the ones obtained for the warehouse without the middle cross aisle for all considered pick-list sizes. This implies that adding the middle cross aisle reduces the average tour length also in the case where picking tours start and end in different locations. This is because the middle cross aisle offers more possibilities for creating tours for the order picker (Roodbergen and de Koster, 2001a). To measure the impact of the middle cross aisle, we calculate the percentage savings in average tour length in both warehouse layouts using the formula $(Z^{wo} - Z^{w})/Z^{wo}$, where Z^{w} and Z^{wo} represent the average tour lengths of the warehouse with and without the middle cross aisle, respectively. It can be seen from Figure 3.11 that the percentage savings in average tour length that result from adding the middle cross aisle to the warehouse are approximately between 16% and 20%. The highest percentage saving was found for a pick-list size of 9 items. It can further be seen that the percentage savings in average tour length get smaller as the pick-list size increases. This is due to the fact that the more items the order picker needs to retrieve in a picking tour, the higher is the chance that the order picker has to traverse all aisles, which reduces the advantage the middle cross aisle brings about. This observation is consistent with the results reported by Roodbergen and de Koster (2001a).

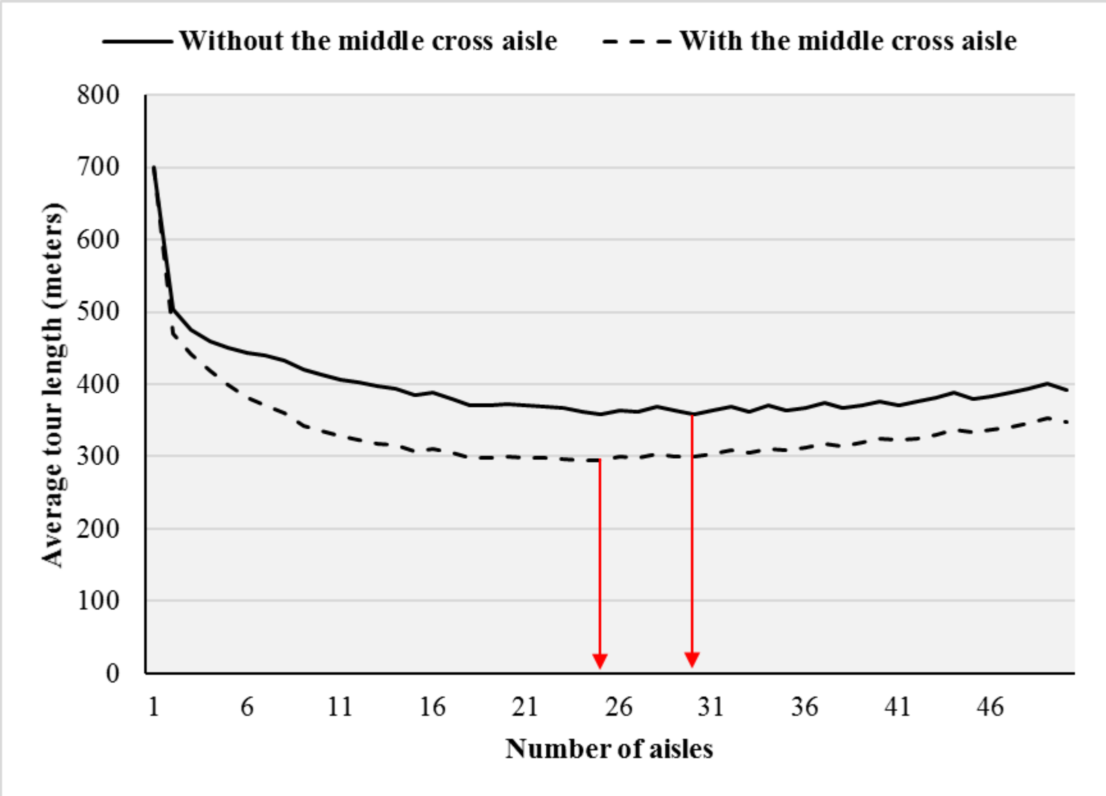


Figure 3.9 Average tour lengths in the warehouse with and without the middle cross aisle for a total aisle length of 450 meters and a pick-list size of 30.

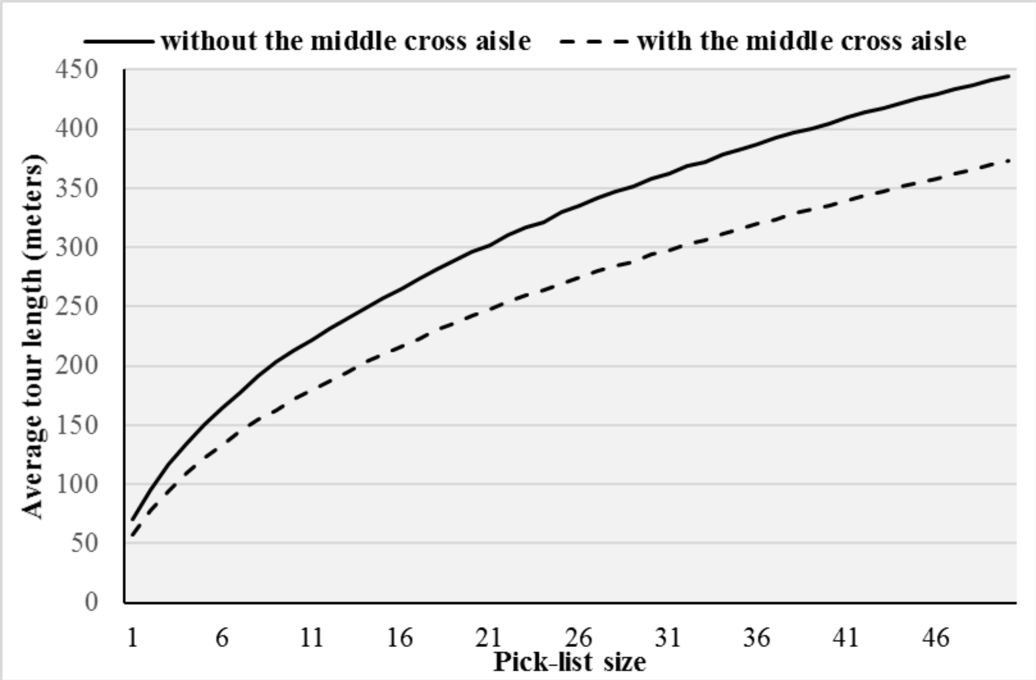


Figure 3.10 Average tour lengths in the warehouse with and without the middle cross aisle for a total aisle length of 450 meters.

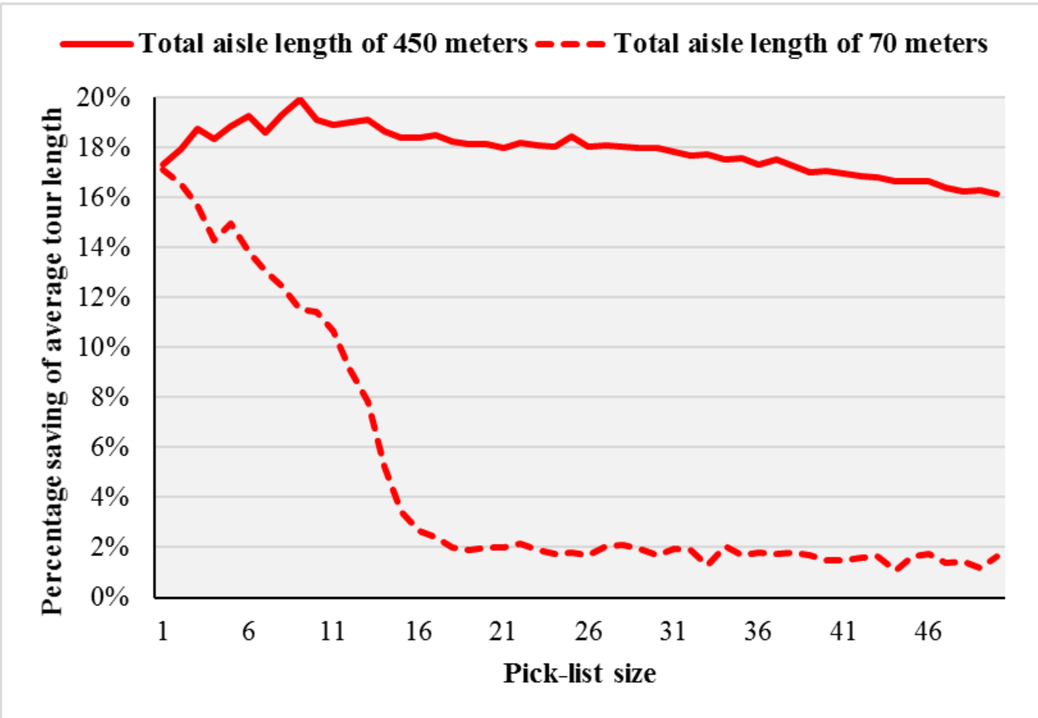


Figure 3.11 Percentage savings in average tour length when a middle cross aisle is added to the warehouse compared to the case without the middle cross aisle for total aisle lengths of 70 and 450 meters.

3.5.2 Comparison of the exact algorithm to a routing heuristic

In this section, we propose a routing heuristic denoted *S*-shape*, and evaluate its performance by comparing the average length of the order picking tours produced by this method to the tours obtained using the exact routing algorithm. The *S*-shape* heuristic can be summarized as follows:

S*-shape: We number the $2r$ picking sub-aisles (each aisle is divided into two sub-aisles; one sub-aisle in each block) from 1 to $2r$ in the following order. Starting in the upper block, the sub-aisles are ordered from left to right. Then we continue to the lower block from right to left, as illustrated in **Figure 3.12**. Assume that sub-aisle x contains the starting point e_1 and sub-aisle y contains the ending point e_2. The order picker starts from e_1, moves to the farthest item in the sub-aisle x, then comes back to the middle cross aisle. S/he then moves along the cross aisle to the sub-aisle $x + 1$ (where sub-aisle $2r + 1$ is sub-aisle 1) if the value of $y - x$ mod $2r$ is among $r, r + 1, \dots, 2r - 1$. On the other hand, s/he proceeds to the sub-aisle $x - 1$ (where sub-aisle 0 is sub-aisle $2r$) if the value of $y - x$ mod $2r$ is among $0,1, \dots, r - 1$. If the sub-aisle is not empty, s/he completely traverses the sub-aisle, then continues to the next sub-aisle determined by the value of $y - x$ as before. If the sub-aisle is empty, s/he skips the sub-aisle, and moves to the next sub-aisle. The process is repeated until s/he has visited all picking locations. In the last step, s/he travels from the last picking location to e_2. **Figure 3.12** shows the routing procedure that results from the *S*-shape* heuristic for the example given in **Figure 3.1**.

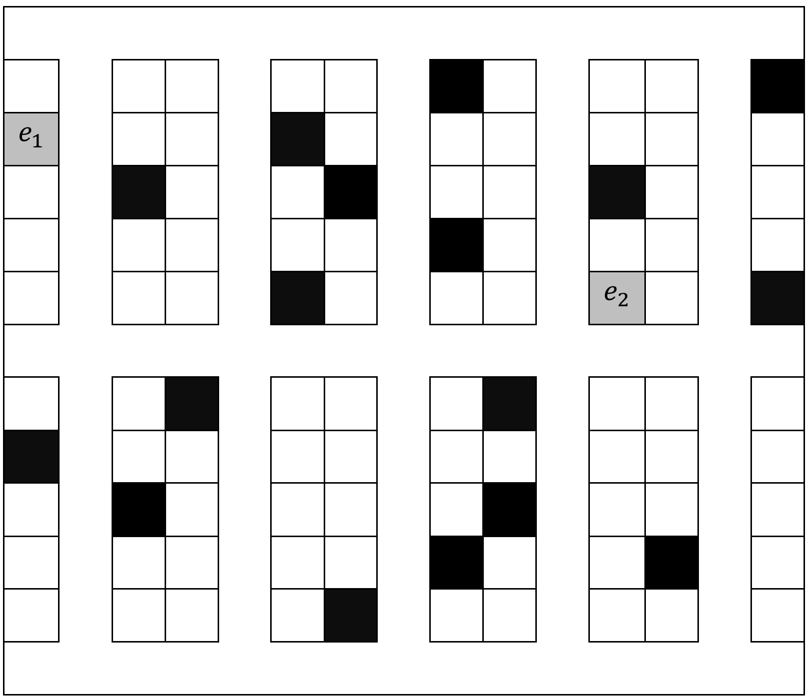


Figure 3.12 The route resulting from the *S*-shape* heuristic for the example from **Figure 3.1**.

We compare the average tour lengths of the *S*-shape* heuristic and the exact algorithm on the optimal two-block warehouse for each pick-list size discussed in Section 3.5.1. **Figure 3.13** and **Figure 3.14** show the average order picking tour lengths obtained from the exact algorithm and the *S*-shape* heuristic for two warehouses with a total aisle length of 450 and 70 meters, respectively. In addition, the percentage optimality gap of the *S*-shape* heuristic is calculated as $(Z^h - Z^e)/Z^e$, where Z^h and Z^e represent the average tour lengths resulting from the *S*-shape* heuristic (*h*) and the exact algorithm (*e*), respectively. The average percentage optimality gaps are presented in **Figure 3.15**. There are a few things to point out from the figures. First, the results demonstrate that the exact algorithm clearly outperforms the *S*-shape* heuristic in all considered cases as expected. Secondly, our results indicate that the optimality gap of the *S*-shape* heuristic is between 6.32% and 35.34% for the warehouse with a total aisle length of 450 meters and between 6.68% and 28.70% for the warehouse with a total aisle length of 70 meters. With respect to the warehouse with a total aisle length of 450 meters, the percentage optimality gap of the *S*-shape* heuristic increases when the pick-list size increases, and it reaches a peak of 35.34% when the pick-list size is 19. After that, the percentage optimality gap slightly decreases to values around 32-34%. Similarly, the percentage optimality gap of the *S*-shape* heuristic for the warehouse with a total aisle length of 70 meters increases when the pick-list size increases and reaches 28.70% when the pick-list size is 23. Afterwards, the percentage optimality gap seems to become stable at around 26-27%.

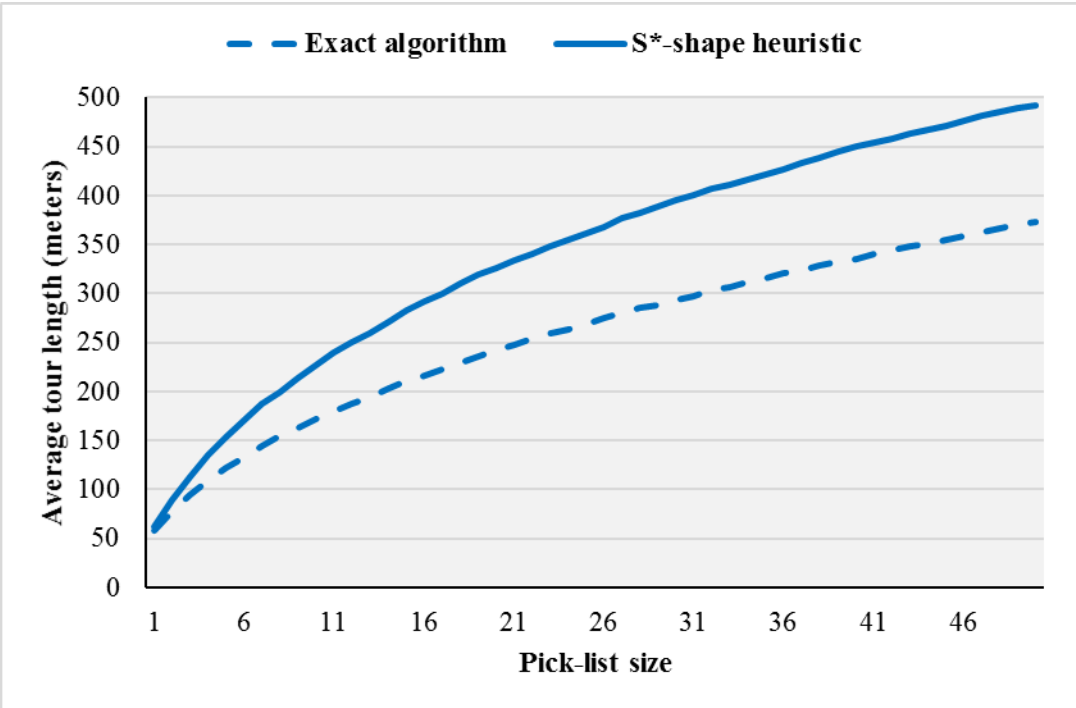


Figure 3.13 Average tour lengths for the exact algorithm and the *S*-shape* heuristic for a warehouse with a total aisle length of 450 meters.

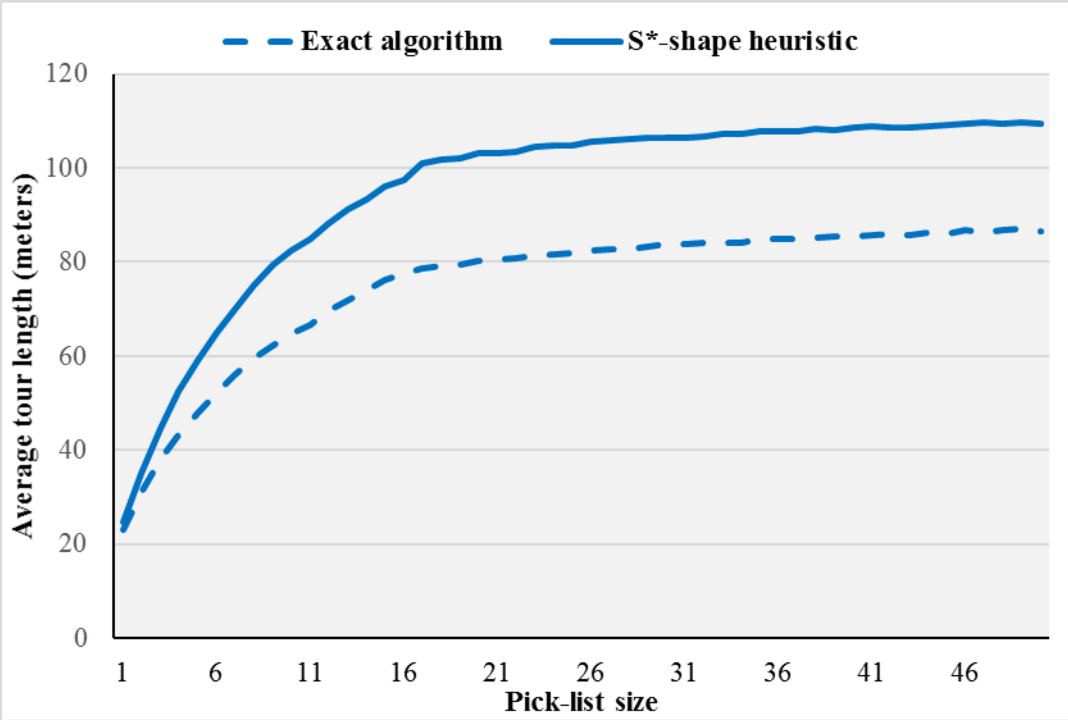


Figure 3.14 Average tour lengths for the exact algorithm and the *S*-shape* heuristic for a warehouse with a total aisle length of 70 meters.

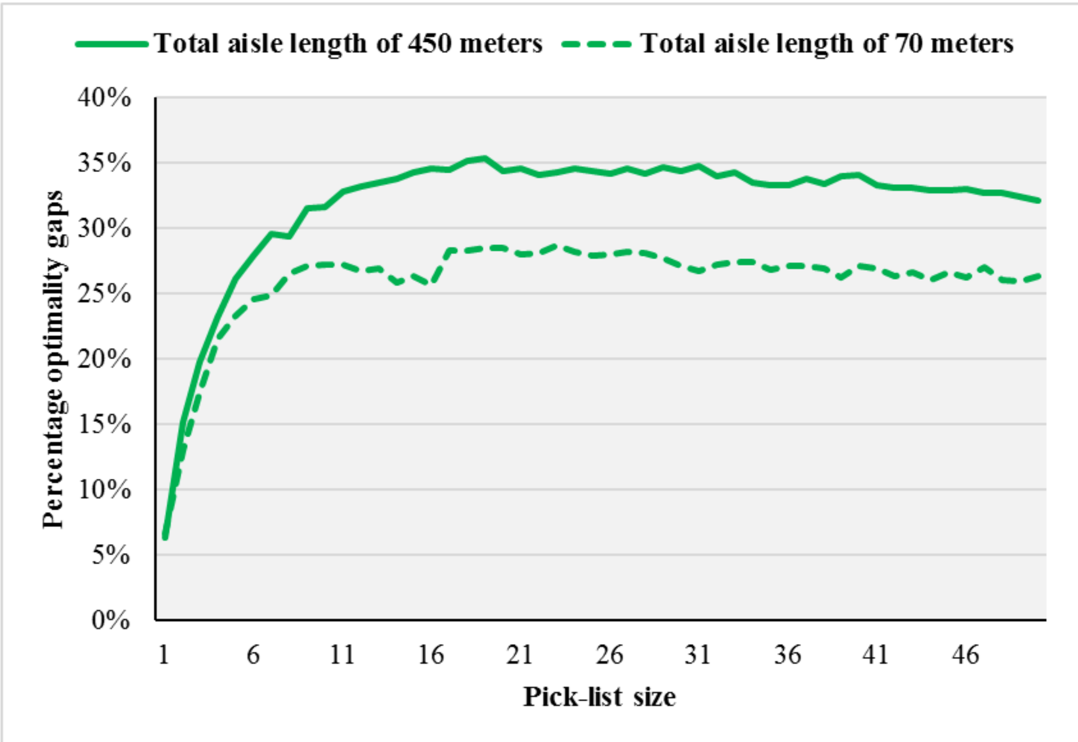


Figure 3.15 Average percentage optimality gaps of the *S*-shape* heuristic for warehouses with total aisle lengths of 70 and 450 meters.

3.6 Conclusion

This chapter proposed an efficient algorithm for determining a minimum-length order picking tour as well as a routing heuristic for a conventional warehouse with two blocks where the starting and ending points of the picking tour are not fixed to the depot, but where they can be any locations in the warehouse instead. This work thus extended an earlier work of Löffler et al. (2018), who studied the case of a conventional warehouse with a single block. The work at hand adapted the solution procedures proposed by RR and Roodbergen and de Koster (2001a), both based on graph theory and dynamic programming, for finding a minimum-length order picking tour for this warehouse layout. An example was presented to illustrate how the proposed algorithm works. Furthermore, we conducted computational experiments to investigate the impact of a middle cross aisle by comparing the average length of the optimal order picking tour in a warehouse with and without a middle cross aisle. Finally, we also compared the performance of the proposed routing heuristic to the exact algorithm. In our experiments, the middle cross aisle reduced the average tour length for every problem setting we studied. Moreover, the exact algorithm obtained tours that were between 6.32% and 35.34% shorter than those generated by the heuristic. These results emphasize that optimal order picker routing should be the preferred means of guiding the order picker through the warehouse.

There are several options for extending this work. First, the chapter at hand assumed that the warehouse has narrow picking aisles, such that the horizontal travel distance of the order picker within an aisle can be neglected. For future research, it would be interesting to extend the present work to a situation where items are retrieved in a wide-aisle warehouse. In this case, it would also be necessary to calculate an additional horizontal travel distance for picking items from both sides of the aisle. Secondly, it was assumed that no order picker congestion occurs within aisles. Relaxing this assumption would be a natural extension of the present work. Moreover, future research could consider turn penalties, which take into account the time lost whenever the order picker enters or leaves an aisle or when a U-turn is necessary within an aisle. Finally, this study could also be extended to other warehouse layouts, e.g., conventional warehouse with multiple blocks, fishbone or flying-V.

Chapter 4 Order picker routing in the chevron warehouse[6]

4.1 Introduction

Chapter 2 revealed that the majority of algorithms that have been proposed for solving the order picker routing problem focused on conventional warehouses (with a single or multiple blocks), accounting for 83% of the algorithms contained in the literature sample. In contrast, the order picker routing problem in non-conventional warehouses has not received much attention so far. Motivated by this observation, this chapter develops both heuristics as well as an algorithm for optimally routing order pickers through a non-conventional warehouse that has been referred to as the chevron warehouse in the literature. The chevron warehouse was originally proposed by Öztürkoğlu et al. (2012) as a new design option for unit-load warehouses with single-command operations. In single-command operations, an order picker travels from the depot to a single location in the warehouse for either storing or retrieving a single pallet, and then returns to the depot (Pohl et al., 2009). To the best of our knowledge, Dukic and Opetuk (2012) is the only work that developed *S-shape* and *composite* routing policies for the chevron warehouse. An optimal order picker routing policy and additional routing heuristics have not been proposed for this warehouse layout so far, and in addition, the performance of the chevron warehouse in terms of order picking time has not been compared to other warehouse layouts yet. Therefore, the main contributions of Chapter 4 are to

1. develop an optimal order picker routing policy and alternative heuristics for this warehouse layout. For developing the optimal routing procedure, we use the concepts of RR and Roodbergen and de Koster (2001a) that are based on graph theory and that utilize a dynamic programming procedure;
2. investigate the effect of different storage assignment policies on the optimal tour;
3. evaluate the performance of the proposed heuristics by comparing them to the proposed exact algorithm for various demand distributions and storage assignment policies used in warehouses;
4. compare the performance of the chevron warehouse to the conventional two-block warehouse based on the average order picking tour lengths obtained by the exact algorithms for various demand distributions and storage assignment policies used in warehouses.

---

[6] Chapter 4 is based on the following paper: Masae, M., Glock, C.H., Vichitkunakorn, P. (2019). Optimal order picker routing in the chevron warehouse. *IISE Transactions*, https://doi.org/10.1080/24725854.2019.1660833.

The remainder of this chapter is structured as follows. The next section describes the problem investigated in this chapter, and Section 4.3 summarizes the works of RR and Roodbergen and de Koster (2001a). Section 4.4 then proposes a procedure for optimally routing order pickers through the chevron warehouse. Section 4.5 introduces the routing heuristics for the chevron warehouse. Extensive computational experiments are presented in Section 4.6. Finally, Section 4.7 concludes the chapter and presents an outlook on future research opportunities.

4.2 Problem description, graph representation and its transformation

The focus of Chapter 4 is on the order picker routing problem in a chevron warehouse, which was presented earlier in Öztürkoğlu et al. (2012).

Figure **4.1** shows an example of such a layout with a single depot in the middle of the front cross aisle. As can be seen, the chevron warehouse consists of three vertical cross aisles, referred to as the left, the middle, and the right vertical cross aisles, which are all perpendicular to the front and back cross aisles. Order pickers can travel from one picking aisle to another through the front, back, and all vertical cross aisles, which do not contain any storage locations. The chevron warehouse is composed of two symmetric parts, referred to as the left and right parts with n picking aisles each. The number of aisles in each part is assumed to be odd since each part of the chevron warehouse is square and symmetric about the longest aisle. d and w denote the distance between two adjacent picking aisles and the width of each storage location, respectively. The example in **Figure 4.1** contains 11 picking aisles in each part. We assume that all picking aisles are narrow, such that order pickers can retrieve the requested items (marked with black boxes in the example in **Figure 4.1**) from storage racks arranged on both sides of the picking aisles without having to cross the aisles. We also assume that order pickers working in the same area can pass each other, which means that we do not consider picker congestion (or picker blocking) within aisles. Note that the first and last aisles of each part only have storage racks on a single side. Moreover, the requested items can be picked directly from the racks without additional vertical travel, which means that we focus on the order picker routing problem in a low-level picker-to-parts system. Furthermore, an item is stored in a single location only, which means we consider order picking in a single storage system. Order picker routing in our case works as follows. The order picker receives a customer order containing a list of items to be picked (a pick-list) at the depot, starts retrieving items from the storage locations until all requested items have been obtained, and then returns to the depot to drop off the retrieved items. The requested items are picked according to the pick-by-order principle, and therefore each round of picking is devoted to a single order. Only a single order picker is considered in the following, who uses a device for transporting retrieved items that has sufficient capacity to accommodate all items requested in the tour.

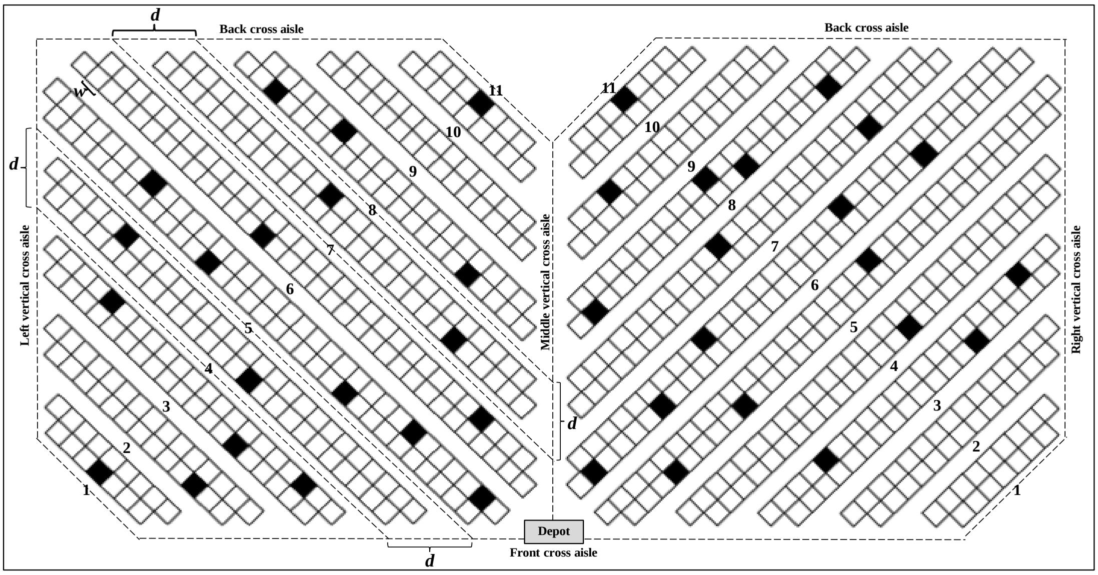


Figure 4.1 Chevron warehouse with 11 picking aisles both in the left and right parts and 40 requested items.

The order picker routing problem can be explained in terms of tours on a graph representation of a warehouse. Using the example in **Figure 4.1**, we define a graph representation $G = (V, E)$ of a chevron warehouse with the set of vertices (V) and edges (E) as shown in **Figure 4.2**. The vertex v_0 represents the location of the depot, whereas the vertices $v_i, i = 1, 2, \dots, m$, are the storage locations of all m items contained on the pick-list. In the left part of the chevron warehouse, we assume that the picking aisles are labeled from the left to right, where the first aisle of the left part is the left-most aisle from the depot. Conversely, for the right part, the picking aisles are labeled from right to left. For $j = 1,2, \dots, n$, the vertices a_j and c_j represent the rear end of aisle j of the left and right parts, respectively. For $j = 1,2, \dots, \frac{n+1}{2}$, the vertices $b_{j,1}$ and $b_{j,2}$ represent the front end of aisle j of the left and right parts, respectively, that intersect the front cross aisle. For $j = \frac{n+1}{2}, \frac{n+3}{2}, \dots, n$, the vertices b_j represents the front end of aisle j of the left and right parts that intersect the middle vertical cross aisle. We note that the vertices $b_{\frac{n+1}{2},1}, b_{\frac{n+1}{2},2}, b_{\frac{n+1}{2}}$ are at the same location, i.e., $b_{\frac{n+1}{2},1} = b_{\frac{n+1}{2},2} = b_{\frac{n+1}{2}}$. **Figure 4.2** shows the graph representation G associated with the order picker routing problem in **Figure 4.1** where $n = 11$ and $m = 40$. For simplicity, the vertex v_0 with zero length is added between b_6 and b_7 (instead of coinciding with b_6). The problem of routing an order picker through a chevron warehouse corresponds to finding a picking tour of minimal length, where a tour is a directed path that visits all requested items in the order of the corresponding graph representation, and vice versa. Hence, the problem of routing an order picker through the warehouse can thus be transformed to the problem of finding a picking tour in the graph representation of the warehouse.

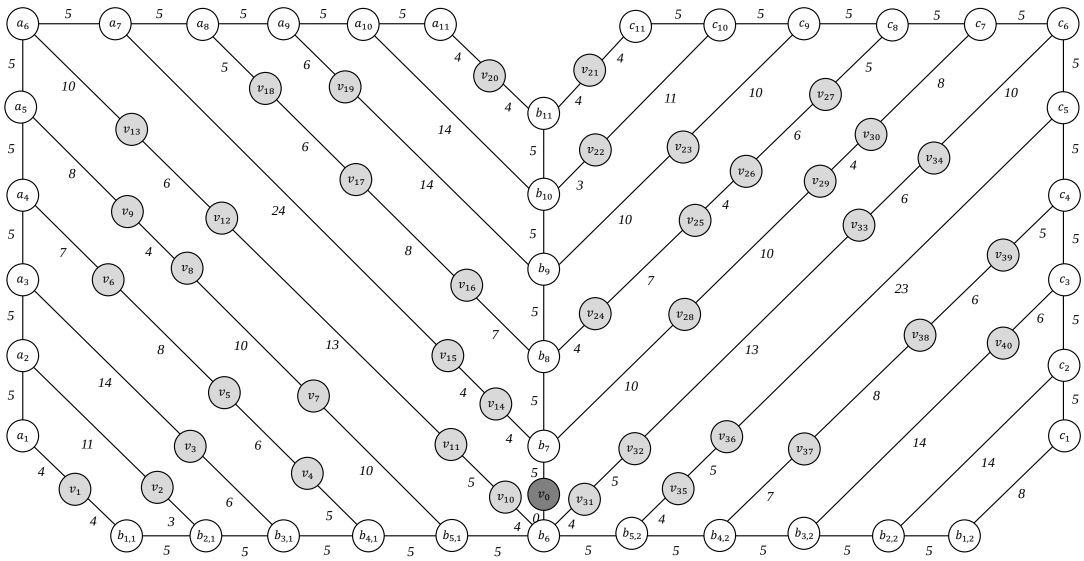
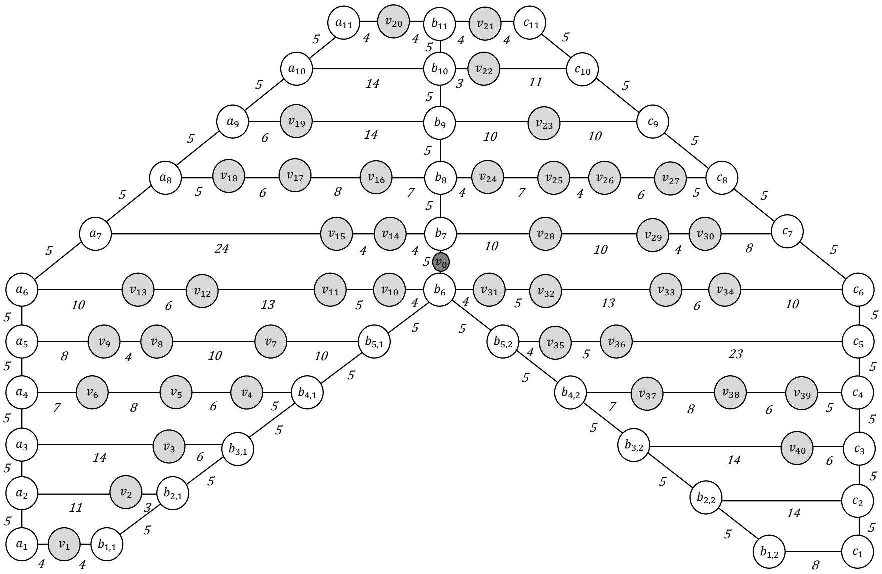
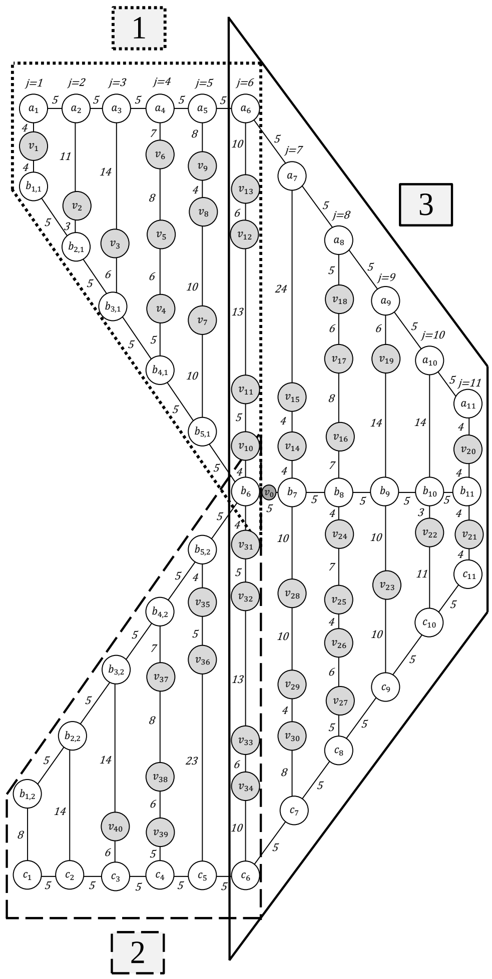


Figure 4.2 Graph representation G of a chevron warehouse with $n = 11$ and $m = 40$.

Çelik and Süral (2014), who studied order picker routing in a fishbone warehouse, transformed the graph representation of the fishbone layout to a graph representation of a conventional warehouse with two blocks as studied earlier by Roodbergen and de Koster (2001a). Motivated by their idea, we transform the graph representation G of a chevron warehouse (**Figure 4.2**) to an equivalent graph, referred to as G^*, as illustrated in Figure 4.3(b). This is done by rotating all parallel aisles in the left part counterclockwise such that they are perpendicular to the middle vertical cross aisle. Similarly, all parallel aisles in the right part are rotated clockwise such that they are perpendicular to the middle vertical cross aisle as shown in Figure 4.3(a). Afterwards, we rotate the graph by another 90 degrees clockwise as illustrated in Figure 4.3(b). We refer to the subgraph consisting of all vertices and edges between a_j and $b_{j,1}$ ($b_{j,2}$ and c_j, a_j and b_j, or b_j and c_j) as the aisle j of the graph. We note that the resulting graph in Figure 4.3(b) consists of three parts, where parts 1 and 2 are each identical to the graph representation of a single-block warehouse as discussed in RR, while part 3 is identical to the graph representation of a two-block warehouse studied in Roodbergen and de Koster (2001a). We can hence adapt the results of RR and Roodbergen and de Koster (2001a) and apply them to our case. The method will be summarized in the following section.



(a)



(b)

Figure 4.3 Graph transformation of the graph representation in Figure 4.2.

4.3 Algorithms of Ratliff and Rosenthal (1983) and Roodbergen and de Koster (2001a)

Since the algorithms of RR and Roodbergen and de Koster (2001a) will be applied in our solution procedure, we briefly summarize them in the following:

With respect to the algorithm of RR, we already summarized it in Section 2.2 of Chapter 2. The reader is referred to this section for more detail. Roodbergen and de Koster (2001a) extended the algorithm of RR to a conventional warehouse with two blocks, referred to as the upper and the lower blocks, as illustrated in **Figure A.15** in the appendix. The time complexity of the extended algorithm is $O(n+m)$, where n and m denote the number of aisles and requested items, respectively. Similar to RR, constructing the graph representation G of such a warehouse is straightforward, as illustrated in **Figure A.16** in the appendix, where the vertices a_j, b_j, and c_j represent the rear end, the middle end, and the front end of aisle j, $j \in 1,2,..,n$, respectively. The authors applied the method of RR to construct a minimum-length tour subgraph by consecutively expanding subgraphs according to the following three transitions: (i) add vertical component in the lower block; (ii) add vertical component in the upper block; and (iii) add horizontal component from the current aisle to the adjacent aisle on the right. For transitions (i) and (ii), vertical components V1-V6 from Figure A.1 are used. For transition (iii), the authors proposed horizontal components H1-H14 connecting two adjacent aisles as shown in **Figure 3.6** in Chapter 3. The following additional definitions are used. Let L_j^- be a subgraph of G containing the vertices a_j, b_j, c_j, and all vertices and edges to the left of them. Let l_j be a subgraph of G containing the vertices b_j, c_j, and all vertices and edges between them, and let $L_j^{+l} = L_j^- \cup l_j$. Similarly, let u_j be a subgraph of G consisting of the vertices a_j and b_j together with all vertices and edges between them, and let $L_j^{+u} = L_j^{+l} \cup u_j$. The necessary and sufficient conditions for a subgraph T_j to be an L_j^-, L_j^{+l}, or L_j^{+u} PTS are listed in Theorem A.2 in the appendix. All L_j^-, L_j^{+l}, or L_j^{+u} PTSs can be grouped by their equivalence classes, which are referred to using a quintuplet (degree parity of a_j, degree parity of b_j, degree parity of c_j, number of connected components, connectivity). The degree parities of a_j, b_j, and c_j can be zero (0), even (E), or uneven (U), whereas the number of connected components can be zero $(0C)$, one $(1C)$, two $(2C)$, or three $(3C)$. The fifth element in the quintuplet, connectivity, only needs to be stated explicitly in case the PTS has two connected $(2C)$ components with even degree each (e.g., $(E,E,E,2C,ab-c)$). In all other cases, the connectivity is straightforward from the degrees of a_j, b_j, and c_j and can be removed from the quintuplet to simplify the notation. Roodbergen and de Koster (2001a) proved that there are 25 equivalence classes to be considered in constructing a minimum-length tour subgraph. The reader is referred to Roodbergen and de Koster (2001a) for more details concerning their equivalence classes.

4.4 Optimal order picker routing in the chevron warehouse

To find the shortest order picking tour on the transformed graph G^* (Figure 4.3(b)), we first construct the minimum-length tour subgraph on G^* by consecutively constructing a minimum-length PTS from aisle $j = 1$ to aisle $j = n$ (see Section 4.4.1). Afterwards, we form the optimal order picking tour from the minimum-length tour subgraph (see Section 4.4.2). To illustrate our proposed algorithm, a numerical example corresponding to the routing problem presented in **Figure 4.1** is presented in Section 4.4.3.

4.4.1 Constructing the minimum-length tour subgraph

The following definition is used throughout the development of the solution procedure.

Definition 4.1 Let G^* be the transformed graph with n picking aisles (labelled from the left-most to the right-most aisles). For $j \in 1,2, \dots, \frac{n+1}{2}$, we let

- $L_{j,1}^{-}$ be the subgraph (in part 1) of G^* containing the vertices a_j and $b_{j,1}$ together with all vertices and edges to the left of them,
- $L_{j,2}^{-}$ be the subgraph (in part 2) of G^* containing the vertices $b_{j,2}$ and c_j together with all vertices and edges to the left of them,
- $m_{j,1}$ be the subgraph (in part 1) of G^* containing the vertices a_j and $b_{j,1}$ together with all vertices and edges between them,
- $m_{j,2}$ be the subgraph (in part 2) of G^* containing the vertices $b_{j,2}$ and c_j together with all vertices and edges between them,
- $L_{j,p}^{+} = L_{j,p}^{-} \cup m_{j,p}$ when $p = 1,2$.

For $j \in \frac{n+1}{2}, \frac{n+3}{2}, \dots, n$, we let

- $L_{j,3}^{-}$ be the subgraph of G^* containing the vertices a_j, b_j, and c_j together with all vertices and edges to the left of them,
- $l_{j,3}$ be the subgraph of G^* containing the vertices b_j and c_j together with all vertices and edges between them,
- $u_{j,3}$ be the subgraph of G^* containing the vertices a_j and b_j together with all vertices and edges between them,
- $L_{j,3}^{+l} = L_{j,3}^{-} \cup l_{j,3}$,
- $L_{j,3}^{+u} = L_{j,3}^{+l} \cup u_{j,3}$.

We note that $L_{\frac{n+1}{2},1}^{+}$ and $L_{\frac{n+1}{2},2}^{+}$ intersect at only a single vertex $b_{\frac{n+1}{2},1} = b_{\frac{n+1}{2},2} = b_{\frac{n+1}{2}}$. To simplify notation, as in RR and Roodbergen and de Koster (2001a), we use L_j to indicate that a result holds if we let $L_j = L_{j,1}^-$, $L_j = L_{j,1}^+$, $L_j = L_{j,2}^-$, or $L_j = L_{j,2}^+$ for $j \in 1,2, \dots, \frac{n+1}{2}$, and $L_j = L_{j,3}^-$, $L_j = L_{j,3}^{+l}$, or $L_j = L_{j,3}^{+u}$ for $j \in \frac{n+1}{2}, \frac{n+3}{2}, \dots, n$.

To find the minimum-length tour subgraph on the transformed graph G^*, we consider a partial tour subgraph of a sequence of increasing subgraphs from aisle $j = 1$ to aisle $j = n$. The definitions of a tour subgraph and a partial tour subgraph are given in Definitions 4.2 and 4.3, respectively.

Definition 4.2 A subgraph T of G^* is a tour subgraph if there is an order picking tour that starts from v_0, passes through the vertices $v_i, i = 1, 2, \dots, m$, and ends at v_0, where each edge in T is traversed exactly once.

Definition 4.3 For any subgraph L_j of G^* (according to Definition 4.1), a subgraph T_j of L_j is an L_j PTS if there exists a subgraph C of G^* consisting of vertices and edges that are contained in G^*, but not in L_j, such that $T_j \cup C$ is a tour subgraph of G^*. The subgraph C is called a completion of the subgraph T_j.

Two L_j PTSs, namely T_j^1 and T_j^2, are said to be equivalent if for any completion C_j of T_j^1 such that $T_j^1 \cup C_j$ is a tour subgraph, $T_j^2 \cup C_j$ is also a tour subgraph, and vice versa. In other words, the set of completions of T_j^1 and T_j^2 coincide. RR and Roodbergen and de Koster (2001a) classified all L_j PTSs by their equivalence classes as mentioned earlier in Section 4.3 and applied a dynamic programming procedure to find the minimum-length tour subgraph by defining the states as the equivalence classes of L_j PTSs and the transitions between states as vertex and edge additions to PTSs.

From Figure 4.3(b), since parts 1 and 2 of the transformed graph G^* are identical to the graph representation of a single-block warehouse, the equivalence classes for all L_j PTSs, $j \in 1,2, \dots, \frac{n+1}{2}$, in parts 1 and 2 are identical to the equivalence classes proposed by RR. In this chapter, the equivalence classes of L_j (either $L_{j,1}^-$ or $L_{j,1}^+$) PTSs in part 1 are referred to by a triple (degree parity of a_j, degree parity of $b_{j,1}$, number of connected components). In part 2, the equivalence classes of L_j (either $L_{j,2}^-$ or $L_{j,2}^+$) PTSs are represented by (degree parity of $b_{j,2}$, degree parity of c_j, number of connected components). Part 3 finally is identical to the graph representation of a two-block warehouse. Thus, the equivalence classes of all L_j PTSs, $j \in \frac{n+1}{2}, \frac{n+3}{2}, \dots, n$, are identical to the equivalence classes proposed by Roodbergen and de Koster (2001a). In our case, the equivalence classes of L_j (either $L_{j,3}^-$, $L_{j,3}^{+l}$, or $L_{j,3}^{+u}$) PTSs in part 3 are classified by a quintuplet (degree parity of a_j, degree parity of b_j, degree parity of c_j, number of connected components, connectivity). Similar to Roodbergen and de Koster (2001a), the variable representing the connectivity is applicable for the case of two connected $(2C)$ components with even degree parity in each a_j, b_j, and c_j.

Our solution procedure contains three main steps. In the first step, the dynamic programming procedure of RR is used to construct the minimum length of $L^+_{\frac{n+1}{2},1}$ and $L^+_{\frac{n+1}{2},2}$ PTSs for each possible equivalence class in parts 1 and 2 of the transformed graph G^*, respectively. Starting with part 1, we construct $L^+_{1,1}$ PTSs for aisle $j = 1$ by adding each vertical component V1-V6 from **Figure A.1** to the empty graph $L^-_{1,1}$. The length of $L^+_{1,1}$ PTSs for each equivalence class are then calculated. Using only the minimum-length $L^+_{1,1}$ PTS in each equivalence class, these $L^+_{1,1}$ PTSs are extended to obtain $L^-_{2,1}$ PTSs by adding each horizontal component H1-H5 from **Figure A.1** to the $L^+_{1,1}$ PTSs. Using only the minimum-length $L^-_{2,1}$ PTS in each equivalence class, we construct $L^+_{2,1}$ PTSs by adding each vertical component in **Figure A.1** to the $L^-_{2,1}$ PTSs. We continue in this fashion for the aisles $j = 3$ to $j = \frac{n+1}{2}$ in part 1 until all $L^+_{\frac{n+1}{2},1}$ PTSs have been obtained. As a result, each $L^+_{\frac{n+1}{2},1}$ PTS is a minimum-length PTS in its own equivalence class, and is a feasible partial order picking tour containing all requested items in part 1. The same procedure is applied to part 2 to obtain the minimum-length $L^+_{\frac{n+1}{2},2}$ PTSs in each equivalence class. We adapt the Pseudo-code presented in Scholz and Wäscher (2017) to summarize the solution procedure for constructing the minimum length of $L^+_{\frac{n+1}{2},p}$ PTSs for $p \in 1,2$ in the following:

Algorithm 1: Pseudo-code for constructing the minimum-length $L^+_{\frac{n+1}{2},p}$ PTSs for $p \in 1,2$.

Input: number of aisles $n_p = \frac{n+1}{2}$, set of storage locations of the requested items in part p.

Output: minimum-length $L^+_{\frac{n+1}{2},p}$ PTS for equivalence classes $(U, U, 1C)$, $(0, E, 1C)$, $(E, 0,1C)$, $(E, E, 1C)$, $(E, E, 2C)$, $(0,0,0C)$, and $(0,0,1C)$.

construct $L^+_{1,p}$ PTSs by adding each vertical component in **Figure A.1** to the empty graph $L^-_{1,p}$;

for equivalence classes $i = 1$ to $i = 7$ in RR **do**

keep the $L^+_{1,p}$ PTS of equivalence class i with the minimum length;

end for

for aisles $j = 2$ to n_p **do**

for equivalence classes $i = 1$ to $i = 7$ **do**

construct $L^{-}_{j,p}$ PTSs by adding each horizontal component in **Figure A.1** to the $L^{+}_{j-1,p}$ PTS of equivalence class i;

end for

for equivalence classes $i = 1$ to $i = 7$ **do**

keep the $L^{-}_{j,p}$ PTS of equivalence class i with the minimum length;

end for

for equivalence classes $i = 1$ to $i = 7$ **do**

construct $L^{+}_{j,p}$ PTSs by adding each vertical component in **Figure A.1** to the $L^{-}_{j,p}$ PTS of equivalence class i;

end for

for equivalence classes $i = 1$ to $i = 7$ **do**

keep the $L^{+}_{j,p}$ PTS of equivalence class i with the minimum length;

end for

end for

for equivalence classes $i = 1$ to $i = 7$ **do**

keep the $L^{+}_{n_p,p}$ PTS with minimum length;

end for

For the second step, each possible pair of equivalence classes of $L^{+}_{\frac{n+1}{2},1}$ PTS and $L^{+}_{\frac{n+1}{2},2}$ PTS are combined, according to Table 4.1, to form an $L^{+u}_{\frac{n+1}{2},3}$ PTS. In other words, the union of an $L^{+}_{\frac{n+1}{2},1}$ PTS and an $L^{+}_{\frac{n+1}{2},2}$ PTS, except only for some cases involving equivalence class $(0,0,1C)$, is interpreted as an $L^{+u}_{\frac{n+1}{2},3}$ PTS as discussed in Theorems 4.1 and 4.2 in the following. Table 4.1 shows the equivalence classes of $L^{+u}_{\frac{n+1}{2},3}$ PTSs that result from the union of a pair of $L^{+}_{\frac{n+1}{2},1}$ and $L^{+}_{\frac{n+1}{2},2}$ PTSs, depending on their equivalence classes. For instance, combining an $(U,U,1C)$ $L^{+}_{\frac{n+1}{2},1}$ PTS with an $(U,U,1C)$ $L^{+}_{\frac{n+1}{2},2}$ PTS will result in an $(U,E,U,1C)$ $L^{+u}_{\frac{n+1}{2},3}$ PTS. This PTS has the uneven-, even-, and uneven-degree parities of $a_{\frac{n+1}{2}}$, $b_{\frac{n+1}{2}}$, and $c_{\frac{n+1}{2}}$, respectively, and a single connected component containing $a_{\frac{n+1}{2}}$, $b_{\frac{n+1}{2}}$, and $c_{\frac{n+1}{2}}$.

Theorem 4.1 Let X be an $L^{+}_{\frac{n+1}{2},1}$ PTS and Y be an $L^{+}_{\frac{n+1}{2},2}$ PTS. Then $X \cup Y$ is an $L^{+u}_{\frac{n+1}{2},3}$ PTS.

Proof. Let X be an $L^{+}_{\frac{n+1}{2},1}$ PTS and Y be an $L^{+}_{\frac{n+1}{2},2}$ PTS. We want to show that $X \cup Y$ is an $L^{+u}_{\frac{n+1}{2},3}$ PTS. First, it is easy to see that $X \cup Y$ is a subgraph of $L^{+u}_{\frac{n+1}{2},3}$. From Theorem A.1 in the appendix, all vertices v_i of $L^{+u}_{\frac{n+1}{2},3}$, representing item locations, are positive in $X \cup Y$. From Table 4.1, it is clear that the conditions (b) and (c) in Theorem 4.3 hold. Hence $X \cup Y$ is an $L^{+u}_{\frac{n+1}{2},3}$ PTS.□

Theorem 4.2 The converse of Theorem 4.1 is also true. If Z is an $L^{+u}_{\frac{n+1}{2},3}$ PTS, then Z can be written as $Z = X \cup Y$, where X is an $L^{+}_{\frac{n+1}{2},1}$ PTS and Y is an $L^{+}_{\frac{n+1}{2},2}$ PTS.

Proof. If Z is an $L^{+u}_{\frac{n+1}{2},3}$ PTS, then there is a subgraph $C_Z \subseteq G^* - L^{+u}_{\frac{n+1}{2},3}$ such that $Z \cup C_Z$ is a tour subgraph. Let $X = Z \cap L^{+}_{\frac{n+1}{2},1}$ and $Y = Z \cap L^{+}_{\frac{n+1}{2},2}$, then we have $Y \cup C_Z$ as a completion of X and $X \cup C_Z$ as a completion of Y. Hence, X is an $L^{+}_{\frac{n+1}{2},1}$ PTS and Y is an $L^{+}_{\frac{n+1}{2},2}$ PTS.□

Theorem 4.3 In a chevron warehouse, a subgraph T is an $L^{+u}_{\frac{n+1}{2},3}$ PTS if and only if all of the following conditions hold:

1. The degrees of all vertices v_i of $L^{+u}_{\frac{n+1}{2},3}$, representing item locations, are positive in T.
2. Every vertex in T, except possibly for $a_{\frac{n+1}{2}}, b_{\frac{n+1}{2}}$, and $c_{\frac{n+1}{2}}$, has even or zero degree.
3. Excluding vertices of zero degree, T has either
   - no connected component,
   - a single connected component containing at least one of $a_{\frac{n+1}{2}}, b_{\frac{n+1}{2}}$, and $c_{\frac{n+1}{2}}$,
   - two connected components, with each component containing at least one of $a_{\frac{n+1}{2}}, b_{\frac{n+1}{2}}$, and $c_{\frac{n+1}{2}}$,
   - three connected components, with each component containing at least one of $a_{\frac{n+1}{2}}, b_{\frac{n+1}{2}}$, and $c_{\frac{n+1}{2}}$.

Proof. Let T be an $L^{+u}_{\frac{n+1}{2},3}$ PTS. By Definition 4.3, there is a completion C such that $T \cup C$ is a tour subgraph. From Euler's theorem, it follows that the conditions (a) and (b) hold. If T contains at least one connected component, each of them must contain at least one of $a_{\frac{n+1}{2}}, b_{\frac{n+1}{2}}$, and $c_{\frac{n+1}{2}}$. Otherwise, $T \cup C$ will not be connected. Hence, the condition (c) holds. Assume now that all conditions hold. If the degrees of $a_{\frac{n+1}{2}}, b_{\frac{n+1}{2}}$, and $c_{\frac{n+1}{2}}$ are all even, we define a completion C to be the subgraph containing all the double edges in $G^* - L^{+u}_{\frac{n+1}{2},3}$. If some of the degrees are odd, we may assume that the degrees of $b_{\frac{n+1}{2}}$ and $c_{\frac{n+1}{2}}$ are odd, while the degree of $a_{\frac{n+1}{2}}$ is even. A completion C can be defined to be the subgraph containing all the double edges in $G^* - L^{+u}_{\frac{n+1}{2},3}$, except for a single edge on a single path through the vertices $b_{\frac{n+1}{2}}$, $b_{\frac{n+3}{2}}$, $c_{\frac{n+3}{2}}$, and $c_{\frac{n+1}{2}}$. As a result, $T \cup C$ is a tour subgraph.□

Table 4.1 Equivalence classes of $L^{+u}_{\frac{n+1}{2},3}$ PTSs that result from the union of all equivalence classes of $L^{+}_{\frac{n+1}{2},1}$ and $L^{+}_{\frac{n+1}{2},2}$ PTSs.

Equivalence classes of $L^{+}_{\frac{n+1}{2},1}$ PTSs	Equivalence classes of $L^{+}_{\frac{n+1}{2},2}$ PTSs						
	$(U,U,1C)$	$(E,0,1C)$	$(0,E,1C)$	$(E,E,1C)$	$(E,E,2C)$	$(0,0,0C)$	$(0,0,1C)$
$(U,U,1C)$	$(U,E,U,1C)$	$(U,U,0,1C)$	$(U,U,E,2C)$	$(U,U,E,1C)$	$(U,U,E,2C)$	$(U,U,0,1C)$	NA
$(E,0,1C)$	$(E,U,U,2C)$	$(E,E,0,2C)$	$(E,0,E,2C)$	$(E,E,E,2C,a-bc)$	$(E,E,E,3C)$	$(E,0,0,1C)$	NA
$(0,E,1C)$	$(0,U,U,1C)$	$(0,E,0,1C)$	$(0,E,E,2C)$	$(0,E,E,1C)$	$(0,E,E,2C)$	$(0,E,0,1C)$	NA
$(E,E,1C)$	$(E,U,U,1C)$	$(E,E,0,1C)$	$(E,E,E,2C,c-ab)$	$(E,E,E,1C)$	$(E,E,E,2C,c-ab)$	$(E,E,0,1C)$	NA
$(E,E,2C)$	$(E,U,U,2C)$	$(E,E,0,2C)$	$(E,E,E,3C)$	$(E,E,E,2C,a-bc)$	$(E,E,E,3C)$	$(E,E,0,2C)$	NA
$(0,0,0C)$	$(0,U,U,1C)$	$(0,E,0,1C)$	$(0,0,E,1C)$	$(0,E,E,1C)$	$(0,E,E,2C)$	$(0,0,0,0C)$	$(0,0,0,1C)$
$(0,0,1C)$	NA	NA	NA	NA	NA	$(0,0,0,1C)$	NA

In the last step, we construct the minimum-length $L^{+u}_{n,3}$ PTSs of each equivalence class in part 3 using the dynamic programming procedure of Roodbergen and de Koster (2001a). First, each horizontal component 1-14 from Figure 3.6 (in Chapter 3) is added to the $L^{+u}_{\frac{n+1}{2},3}$ PTSs, resulting in $L^{-}_{\frac{n+3}{2},3}$ PTSs. The $L^{-}_{\frac{n+3}{2},3}$ PTS with minimum length in each equivalence class is kept, while the others are removed from the set of candidate solutions for the optimal solution. Each vertical component V1-V6 from **Figure A.1** is then added to the $L^{-}_{\frac{n+3}{2},3}$ PTSs in the lower block (between $b_{\frac{n+3}{2}}$ and $c_{\frac{n+3}{2}}$) to obtain $L^{+l}_{\frac{n+3}{2},3}$ PTSs. Using only the minimum-length $L^{+l}_{\frac{n+3}{2},3}$ PTSs in each equivalence class, we construct $L^{+u}_{\frac{n+3}{2},3}$ PTSs by adding each vertical component from **Figure A.1** to the $L^{+l}_{\frac{n+3}{2},3}$ PTSs in the upper block (between $a_{\frac{n+3}{2}}$ and $b_{\frac{n+3}{2}}$). Lastly, the minimum-length $L^{+u}_{\frac{n+3}{2},3}$ PTS in each equivalence class is kept, and the algorithm continues in this fashion for the next aisles $j = \frac{n+5}{2}$ to $j = n$ until we obtain all $L^{+u}_{n,3}$ PTSs. The minimum-length tour subgraph is the shortest PTS among the equivalence classes $(0,0,0,1C)$, $(E,0,0,1C)$, $(0,E,0,1C)$, $(0,0,E,1C)$, $(E,E,0,1C)$, $(E,0,E,1C)$, $(0,E,E,1C)$, and $(E,E,E,1C)$. The solution procedure for constructing the minimum-length tour subgraph on G^* can be summarized as follows. In terms of time complexity, the proposed algorithm is linear in the number of requested items m and picking aisles n, i.e. $O(m+n)$.

Algorithm 2: Pseudo-code for constructing the minimum-length tour subgraph on G^*.
Input: number of aisles n, set of storage locations of the requested items in G^*.
Output: the minimum-length tour subgraph on G^*.

construct the minimum-length $L^{+}_{\frac{n+1}{2},p}$ PTSs for $p \in 1,2$ using **Algorithm 1**.
combine each possible pair of equivalence classes of $L^{+}_{\frac{n+1}{2},1}$ PTS and $L^{+}_{\frac{n+1}{2},2}$ PTS to form an $L^{+u}_{\frac{n+1}{2},3}$ PTS (see Table 4.1);
for equivalence classes $i = 1$ to $i = 25$ in Roodbergen and de Koster (2001a) **do**
    keep the $L^{+u}_{\frac{n+1}{2},3}$ PTS of equivalence class i with minimum length;
end for
for aisles $j = \frac{n+3}{2}$ to n **do**
    for equivalence classes $i = 1$ to $i = 25$ in Roodbergen and de Koster (2001a) **do**
        construct $L^{-}_{j,3}$ PTSs by adding each horizontal component in Figure 3.6 to the $L^{+u}_{\frac{n+1}{2},3}$ PTS of equivalence class i;
        keep the $L^{-}_{j,3}$ PTS of equivalence class i with the minimum length;
    end for
    for equivalence classes $i = 1$ to $i = 25$ **do**
        construct $L^{+l}_{j,3}$ PTSs by adding each vertical component in **Figure A.1** to the $L^{-}_{j,3}$ PTS of equivalence class i;
        keep the $L^{+l}_{j,3}$ PTS of equivalence class i with the minimum length;
    end for
    for equivalence classes $i = 1$ to $i = 25$ **do**
        construct $L^{+u}_{j,3}$ PTSs by adding each vertical component in **Figure A.1** to the $L^{+l}_{j,3}$ PTS of equivalence class i;
        keep the $L^{+u}_{j,3}$ PTS of equivalence class i with the minimum length;
    end for

end for

out of equivalence classes $(0,0,0,1C)$, $(E,0,0,1C)$, $(0,E,0,1C)$, $(0,0,E,1C)$, $(E,E,0,1C)$, $(E,0,E,1C)$, $(0,E,E,1C)$, and $(E,E,E,1C)$, determine the minimum-length $L_{n,3}^{+u}$ PTS, which is the minimum-length tour subgraph on G^*.

4.4.2 Tour construction algorithm

In Section 4.4.1, we outlined a procedure for finding the minimum-length tour subgraph on the transformed graph G^* that contains the locations of all items requested in the order as well as the depot. This section presents the general steps of the procedure for constructing the minimum-length order picking tour in the warehouse from the minimum-length tour subgraph. We use the tour construction procedure presented earlier by RR to solve our problem. The procedure can be summarized as follows:

Step 1. Start the order picking tour at the depot v_0 as the first vertex visited.

Step 2. Let v^* be the vertex currently being visited.

Step 3. If there is a pair of unused parallel edges incident to v^*, choose one of them to move to the next vertex, then go back to *Step 2*. Otherwise, continue to the next step.

Step 4. If there are unused single edges that are not a pair of parallel edges from *Step 3*, incident to v^*, choose one of them to move to the next vertex, then go back to *Step 2*. Otherwise, continue to the next step.

Step 5. If there is a pair of parallel edges incident to v^* including an unused edge, choose it to move to the next vertex, then go back to *Step 2*. Otherwise, continue to the next step.

Step 6. The order picking tour ends at the depot v_0 as the last vertex visited.

4.4.3 Numerical example

This section illustrates our solution procedure using the example presented in **Figure 4.1** with $n = 11$. Our goal is to construct the minimum-length tour subgraph of the transformed graph G^* and to use it to form the optimal order picking tour in the warehouse. As mentioned earlier in Section 4.4.1, there are three main steps in the construction of the minimum-length tour subgraph on G^*:

Step 1: Construct the minimum-length $L_{6,1}^{+}$ and $L_{6,2}^{+}$ PTSs.

To find the minimum-length $L_{6,1}^{+}$ PTSs in part 1 of G^*, we use **Algorithm 1** by first constructing $L_{1,1}^{+}$ PTSs by simply adding each of the vertical component V1-V5 from **Figure A.1** to the empty graph $L_{1,1}^{-}$. The resulting $L_{1,1}^{+}$ PTSs are of equivalence classes $(U,U,1C)$, $(E,0,1C)$, $(0,E,1C)$, $(E,E,2C)$, and $(E,E,1C)$ with the minimum lengths 8, 8, 8, 8, and 16, respectively. From $L_{1,1}^{+}$ PTSs to $L_{2,1}^{+}$ PTSs, we do the following: Firstly, $L_{2,1}^{-}$ PTSs are constructed by adding each horizontal component H1-H4 from **Figure A.1** to the resulting $L_{1,1}^{+}$

PTSs from the previous transition. For each equivalence class, only an $L_{2,1}^{-}$ PTS of minimum length is selected and used in the subsequent steps. We repeat this process to obtain all $L_{3,1}^{+}, L_{4,1}^{+}, L_{5,1}^{+}$ and $L_{6,1}^{+}$ PTSs. Figure 4.4(a) shows a minimum-length $L_{6,1}^{+}$ PTS of equivalence class $(E, E, 1C)$ obtained from our procedure. Similarly, **Algorithm 1** is used to construct the minimum-length $L_{6,2}^{+}$ PTSs in part 2 of G^{*}. A minimum-length $L_{6,2}^{+}$ PTS of equivalence class $(E, E, 1C)$ obtained from our procedure is shown in Figure 4.4(b).

Step 2: Combine each possible pair of $L_{6,1}^{+}$ and $L_{6,2}^{+}$ PTSs.

In this step, each pair of $L_{6,1}^{+}$ and $L_{6,2}^{+}$ PTSs from the previous step are combined to form an $L_{6,3}^{+u}$ PTS. Figure 4.5 shows an instance of an $L_{6,3}^{+u}$ PTS that results from the $L_{6,1}^{+}$ PTS and the $L_{6,2}^{+}$ PTS from Figure 4.4(a) and Figure 4.4(b), respectively.

Step 3: Construct the minimum-length $L_{11,3}^{+u}$ PTSs.

To obtain the minimum-length $L_{11,3}^{+u}$ PTSs, we apply the algorithm of Roodbergen and de Koster (2001a) by first adding each horizontal component 1-14 from Figure 3.6 (in Chapter 3) to the $L_{6,3}^{+u}$ PTSs, resulting in $L_{7,3}^{-}$ PTSs. After that, each vertical component V1-V6 from **Figure A.1** is added to the $L_{7,3}^{-}$ PTSs in the lower block (between b_7 and c_7) to obtain $L_{7,3}^{+l}$ PTSs. We further construct $L_{7,3}^{+u}$ PTSs by adding each vertical component from **Figure A.1** to the $L_{7,3}^{+l}$ PTSs in the upper block (between a_7 and b_7). The same procedure is applied to the next aisles. In the end, all $L_{11,3}^{+u}$ PTSs are obtained. Figure 4.6 shows a minimum-length $L_{11,3}^{+u}$ PTS of equivalence class $(E, E, 0,1C)$, which is also the minimum-length tour subgraph of G^{*}.

Lastly, we can transform the resulting subgraph of G^{*} in Figure 4.6 back to a subgraph of G as shown in Figure 4.7. The tour construction procedure presented in Section 4.4.2 is then used to obtain a minimum-length order picking tour in the warehouse.

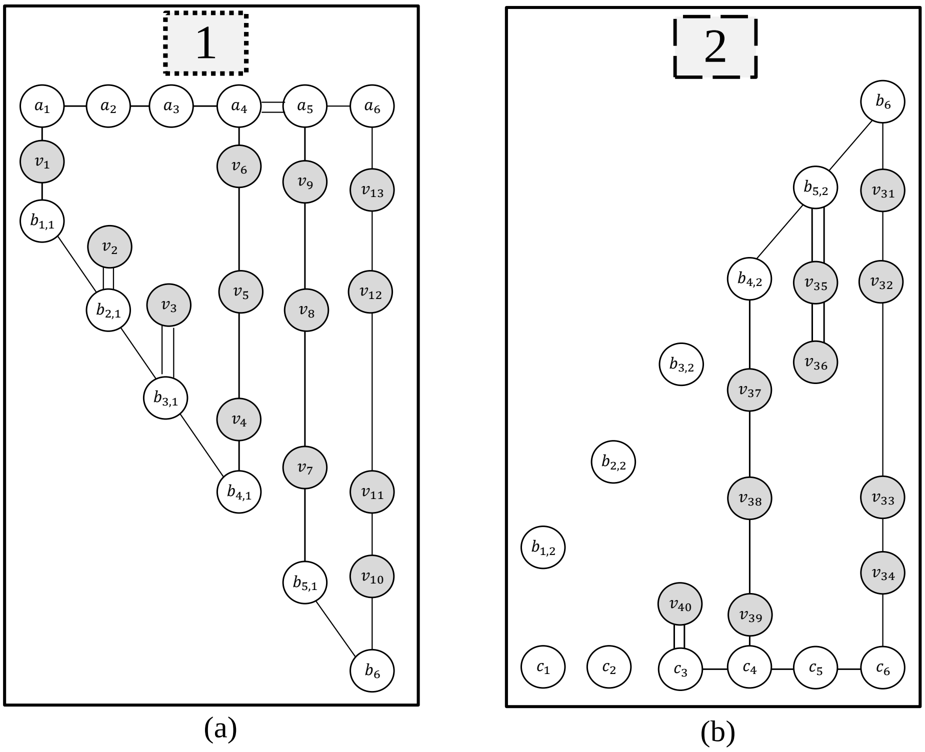


Figure 4.4 The minimum-length $L^+_{6,1}$ (a) and $L^+_{6,2}$ (b) PTSs of equivalence class $(E, E, 1C)$.

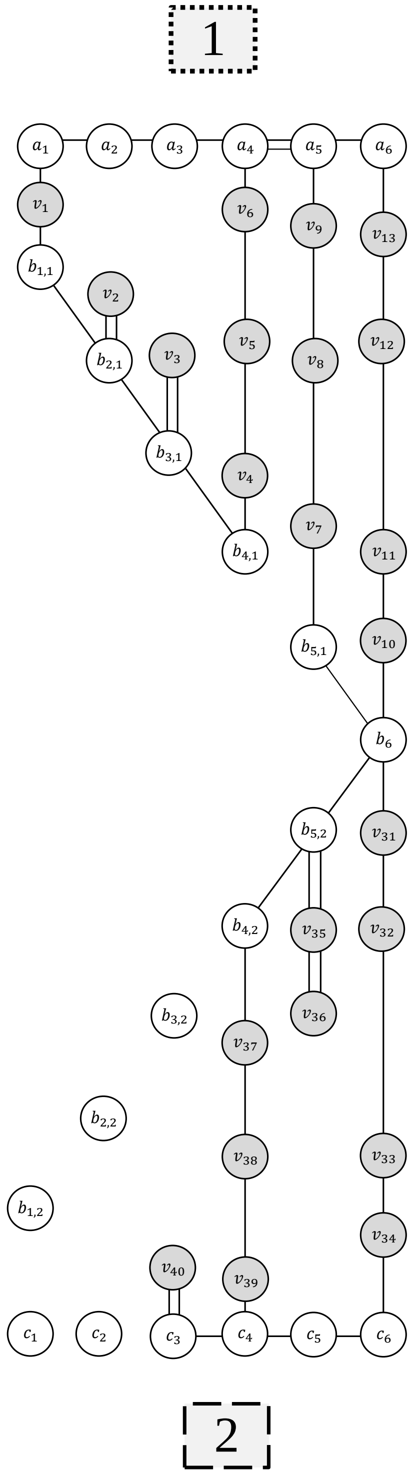


Figure 4.5 The minimum-length $L_{6,3}^{+u}$ PTS of equivalence class $(E, E, E, 1C)$.

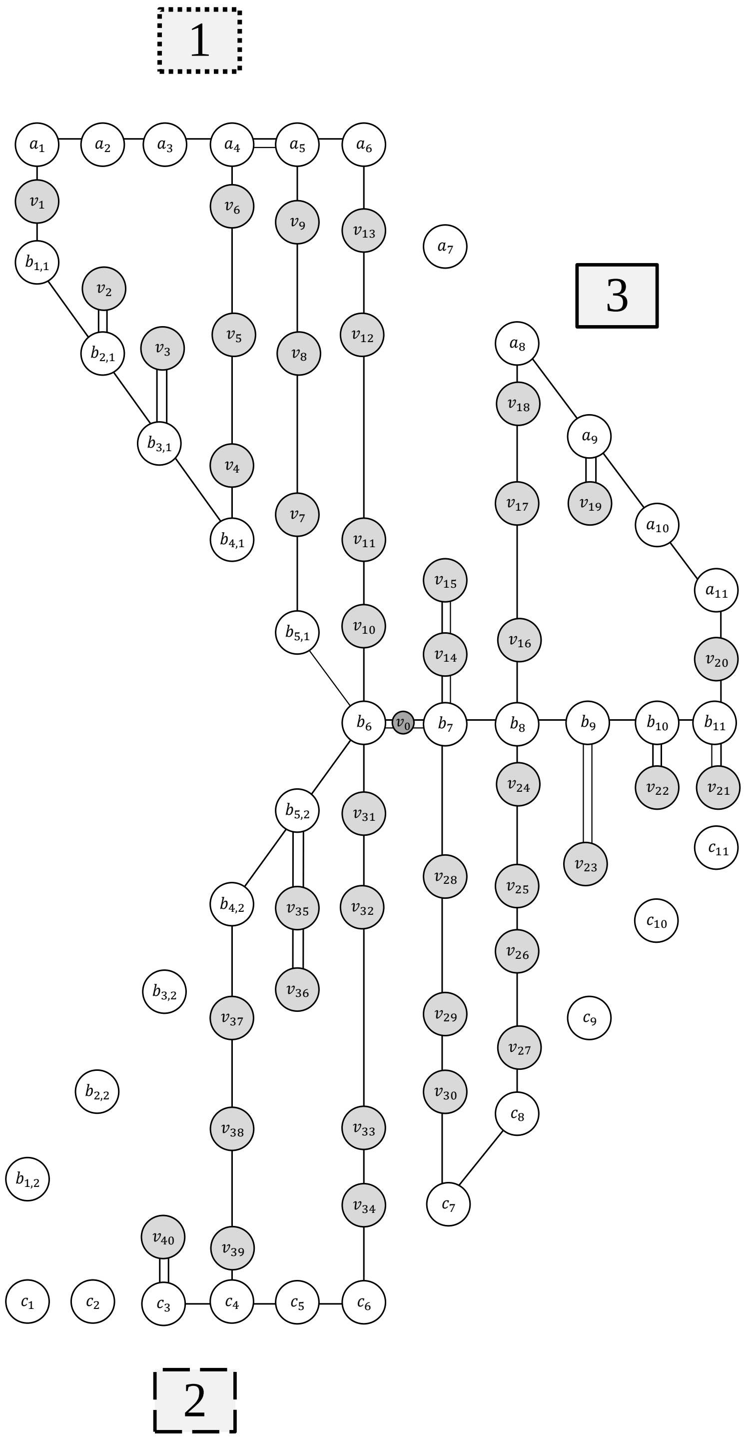


Figure 4.6 The minimum-length $L_{11,3}^{+u}$ PTS of equivalence class $(E, E, 0,1C)$.

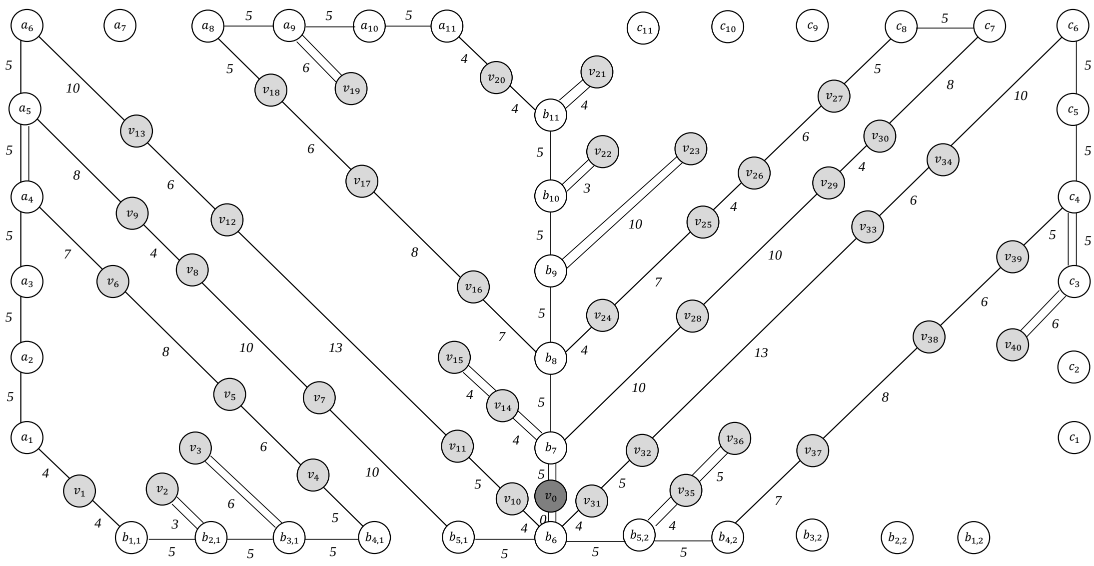


Figure 4.7 The minimum-length tour subgraph of the graph representation G from **Figure 4.2**.

4.5 Routing heuristics for the chevron warehouse

Dukic and Opetuk (2012) proposed the *chevron S-shape heuristic* for guiding an order picker through a chevron warehouse by treating each part of the chevron warehouse as a single-block warehouse. Motivated by their ideas, we propose two additional heuristics, namely *chevron midpoint* and *chevron largest gap*. The performance of these two heuristics as well as the *chevron S-shape* heuristic are then compared to the exact algorithm presented in the previous section. The proposed heuristics and the *chevron S-shape* heuristic can be summarized as follows:

Chevron midpoint: This heuristic divides aisles in each part of the chevron warehouse into two equal halves, referred to as the upper and the lower sections. The order picker starts at the depot, moves to the first aisle on the left that contains requested items in the lower section, and picks the requested items by entering and leaving the aisle from the same side without accessing the upper section. Afterwards, s/he moves to the next aisles containing requested items and retrieves them in the same fashion. The left-most aisle containing requested items is completely traversed, and then the order picker repeats the picking process in the upper sections of the aisles in the left part of the warehouse. S/he traverses the last aisle containing requested items in the upper sections and uses the middle vertical cross aisle to retrieve all remaining requested items in the left part. The order picker uses the same picking method for the right part of the chevron warehouse and finally returns to the depot. **Figure 4.8** shows the routing procedure that results from the *chevron midpoint* heuristic for the example given in **Figure 4.1**.

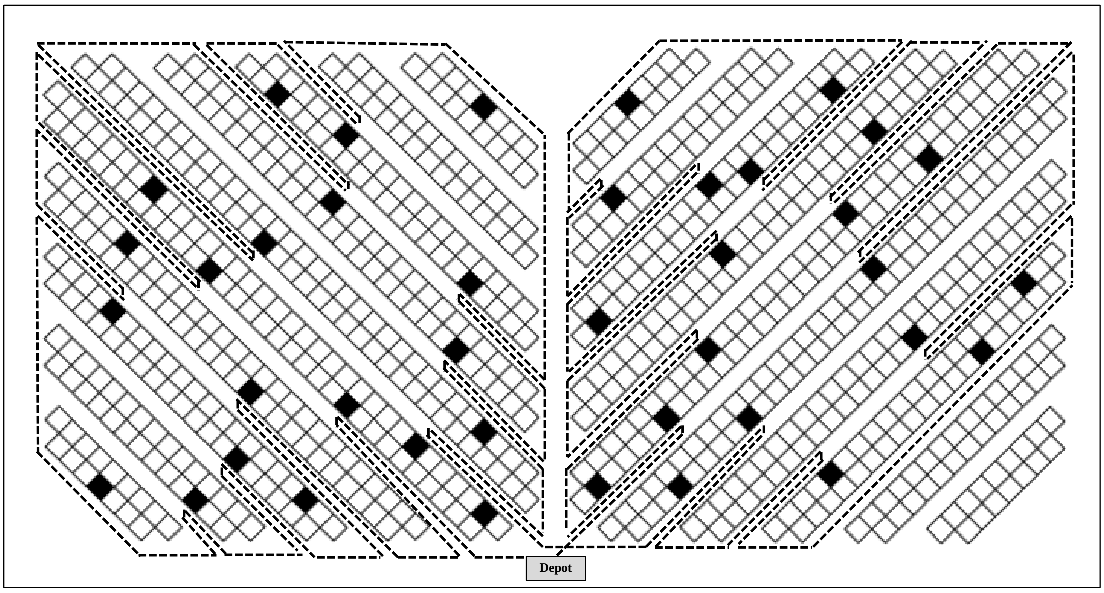


Figure 4.8 The route resulting from the *chevron midpoint* heuristic for the example from **Figure 4.1**

Chevron largest gap: This heuristic is similar to *chevron midpoint* except that each aisle is divided into an upper and a lower section using the largest gap between two requested items in the aisle or between the aisle exits and a requested item. As in the case of *chevron midpoint*, the order picker applies the same procedure to retrieve all requested items in the left part of the warehouse and then moves to the right part to collect requested items there. Figure 4.9 shows the route resulting from the *chevron largest gap* heuristic for the example given in **Figure 4.1**. Note that **Figure 4.8** and Figure 4.9 differ in picking aisle 8 of the right part of the chevron warehouse.

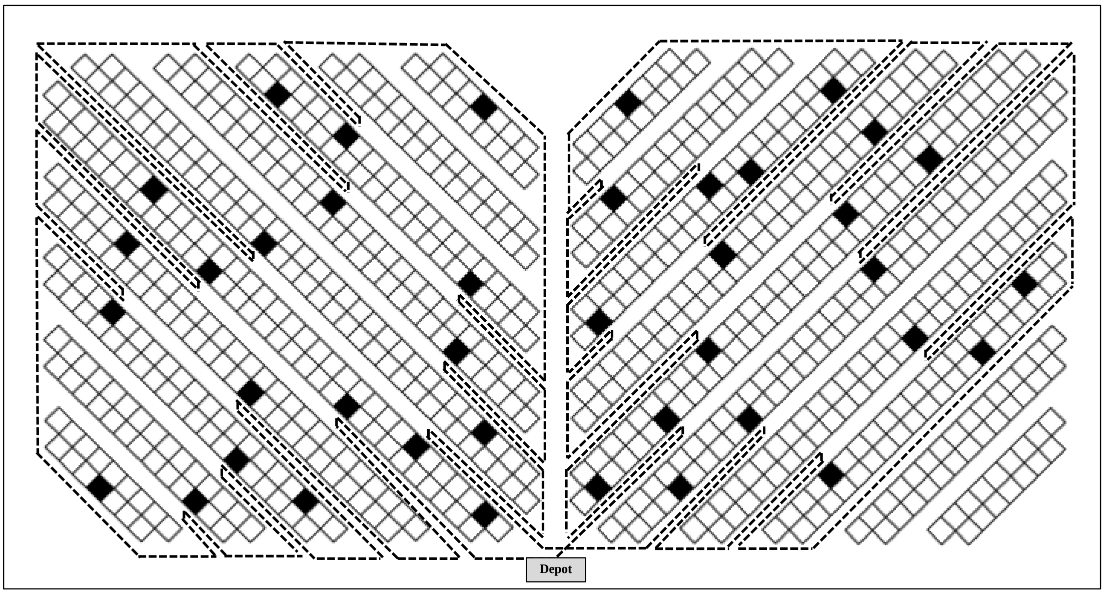


Figure 4.9 The route resulting from the *chevron largest gap* heuristic for the example from Figure 4.1.

Chevron S-shape: The order picker starts by moving from the depot to the left-most aisle of the left part that contains requested items and traverses the aisle completely. S/he moves to the next aisle in the same part that contains requested items, traverses this aisle completely, and continues in this fashion until all requested items in the left part have been retrieved. S/he then moves to the right part, picks requested items according to the same principle from the back- to the front-most aisles containing requested items, and returns to the depot. Figure 4.10 presents the *chevron S-shape* solution for the example given in **Figure 4.1**.

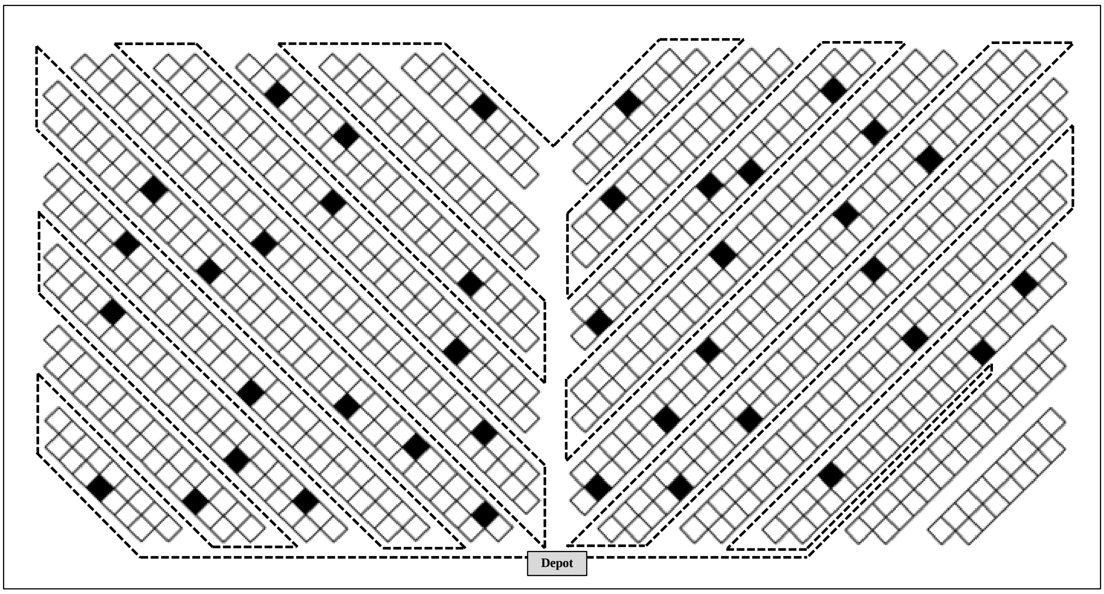


Figure 4.10 The route resulting from the *chevron S-shape* heuristic for the example from Figure 4.1.

4.6 Computational experiments

In computational experiments, we first investigate the effect of different storage assignment policies on the optimal tour. Secondly, we evaluate the performance of the proposed heuristics compared to the exact algorithm. Thirdly, we compare the performance of the chevron warehouse to the conventional two-block warehouse (depicted in Figure 4.11) based on the average order picking tour lengths obtained by the exact algorithms. We use the parameters summarized in Table 4.2 for the purpose of our experiments, with some problem sets taken from Çelik and Süral (2014). As can be seen, the order picker routing policies investigated here include the *optimal (O)*, the *chevron midpoint (CM)*, the *chevron largest gap (CL)*, and the *chevron S-shape (CS) policy*. The storage assignment policies considered are random storage (with uniform demand) and turnover-based storage with 20/40, 20/60, and 20/80 demand skewness, where the pattern x/y means that x% of the items (or stock keeping units (SKUs)) account for y% of the total demand. We consider nine different chevron warehouse sizes with 3, 5, 7, 9, 11, 13, 15, 17, and 19 picking aisles in each part, resulting in 6, 10, 14, 18, 22, 26, 30, 34, and 38 picking aisles with a total of 64, 176, 336, 544, 800, 1104, 1456, 1856, and 2304 number of storage locations, respectively. They are referred to as WZ1, WZ2, WZ3, WZ4, WZ5, WZ6, WZ7, WZ8, and WZ9, respectively. For each warehouse, we assume that different SKUs are located in the different storage locations. For each instance, we vary pick-list sizes (number of items in an order) as 1, 2, 3, 5, 10, 30, and 60 to investigate their effects on the relative performance of the order picker routing policies.

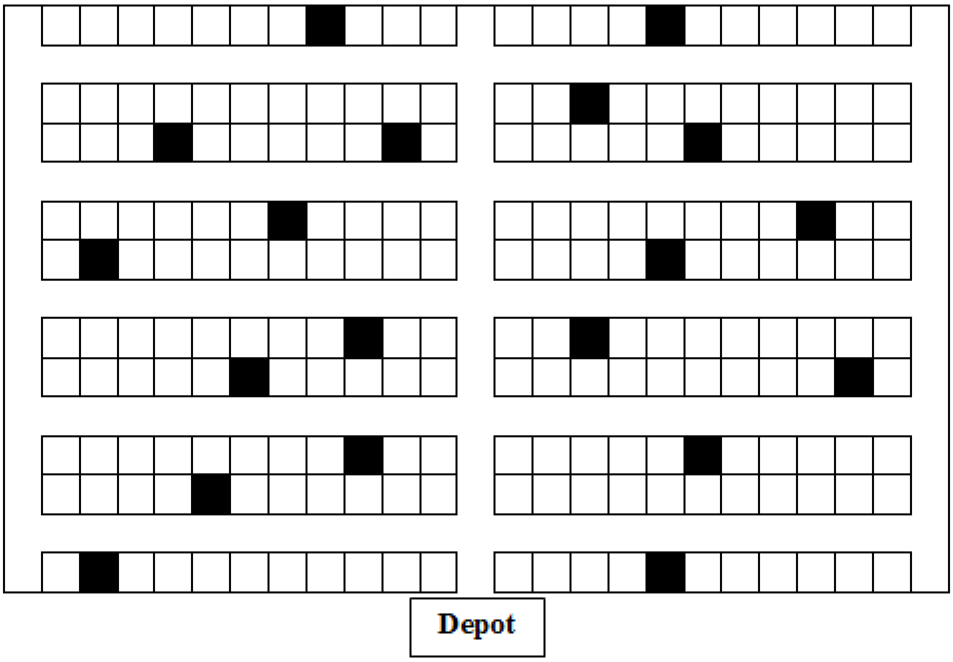


Figure 4.11 Conventional warehouse with two blocks.

Table 4.2 Parameters used for evaluating the performance of the proposed order picker routing policies**.**

Routing strategies	*O, CM, CL, CS*
Storage assignment policies	Random storage, Turnover-based storage (20/40, 20/60, 20/80)
Warehouse sizes	WZ1, WZ2, WZ3, WZ4, WZ5, WZ6, WZ7, WZ8, WZ9
Pick-list sizes	1, 2, 3, 5, 10, 30, 60

For each instance, we randomly generate 1,000 orders to evaluate the average order picking tour length. In our experiments, we set the distance between two neighboring storage locations and two adjacent picking aisles to 1 and 5 meter(s) (or 3.28 and 16.40 feet), respectively. The distance from the respective last picking positions in the picking aisles to a cross aisle is set to 1 meter (3.28 feet), which means the order picker has to walk 1 meter to leave an aisle. All routing procedures were implemented in Java, and all instances were run on a computer with Intel Core i5-7200U 2.50 GHz and 8 GB RAM.

Table A.2 (in the appendix) gives an overview of the average order picking tour lengths of the exact algorithms in combination with (i) random storage with uniform demand, (ii) turnover-based storage with 20/40 demand skewness, (iii) turnover-based storage with 20/60 demand skewness, and (iv) turnover-based storage with 20/80 demand skewness. With respect to turnover-based storage with demand skewness, we heuristically adapted the storage assignment policy proposed by Pohl et al. (2011) and Çelik and Süral (2014) by assigning a higher probability of demand to items closer to the depot. Firstly, the storage locations are sorted in increasing order of their distance from the depot. Secondly, to account for demand skewness, the model of Bender (1981) is used to determine the probability of demand of each item stored in the storage location i. Bender's model uses an analytical function of the form

$$\mathrm{F}(x) = \frac{(1+A)x}{A+x},$$

where A is a shape factor depending on the demand skewness. In this chapter, we consider three demand skewness patterns, namely 20/40, 20/60, and 20/80, where the values of A are 0.60, 0.20, and 0.07, respectively. We assume that the number of items in the warehouse is N. The probability p_i of demand for item i can be calculated from

$$p_i = F\left(\frac{i}{N}\right) - F\left(\frac{i-1}{N}\right), \text{ for } i = 1,2,\ldots,N.$$

As a result, an item closer to the depot now has a higher probability of demand.

We summarize the effect of the four storage assignment policies on the average optimal tour length in Table A.2, Figure 4.12, and Figure 4.13. From Figure 4.12, it is obvious that the average tour length decreases when the skewness of demand increases. This is due to the fact that items with higher demand are assigned to the aisles closest to the depot, resulting in shorter travel distances required for retrieving these items. In addition, the figure suggests that the average tour length grows almost linearly in the number of aisles. However, the number of storage locations in our warehouse is quadratic in the number of aisles. Hence, the average tour length grows in the number of locations according to the power of $1/2$. Moreover, it can be seen that the average tour length in warehouses using a turnover-based storage with a 20/80 demand skewness gradually increases when the warehouse size increases. Conversely, it rapidly increases in warehouses using a random storage policy. To obtain the possible savings in tour length produced by the exact algorithm under turnover-based storage as compared to the case of random storage, we calculated the percentage gap between the average tour length obtained by these two storage policies using the following formula:

$$Gap = \frac{Z^{tu} - Z^{ra}}{Z^{ra}},$$

where Z^{ra} and Z^{tu} represent the average tour length obtained by the exact algorithm in combination with random (*ra*) and turnover-based storage assignment (*tu*) policies, respectively. All gaps resulting in our experiment are negative, which indicates that the average tour length for turnover-based storage assignment is shorter than that obtained for random storage assignment. For convenience, we define the improvement as the negative value of the gap, which is now positive. Then, a higher improvement means that the average tour length is lower for the turnover-based policy as compared to the random assignment. Figure 4.13 shows the percentage improvement of turnover-based storage over random storage. As can be seen, turnover-based storage policies with 20/40, 20/60, and 20/80 demand skewness are between

11.91-12.71%, 22.53-27.36%, and 32.66-45.16% better than random storage. When the chevron warehouse has a larger number of aisles, say more than 22 aisles or equivalently more than 1,000 locations, the percentage improvements of turnover-based storage with 20/40, 20/60, and 20/80 demand skewness are stable around 13%, 27%, and 45%, respectively. It is worth pointing out that these improvements are the average over seven pick-list sizes. Indeed, Table A.2 indicates that the improvement strongly depends on the pick-list size. For example, in warehouse WZ9, the percentage improvements for turnover-based storage with 20/40, 20/60, and 20/80 demand skewness are around 23%, 39%, and 55% when the pick-list size is 1; they are only 6%, 19%, and 37% when the pick-list size is 60. This suggests that an increase in pick-list size leads to a lower improvement.

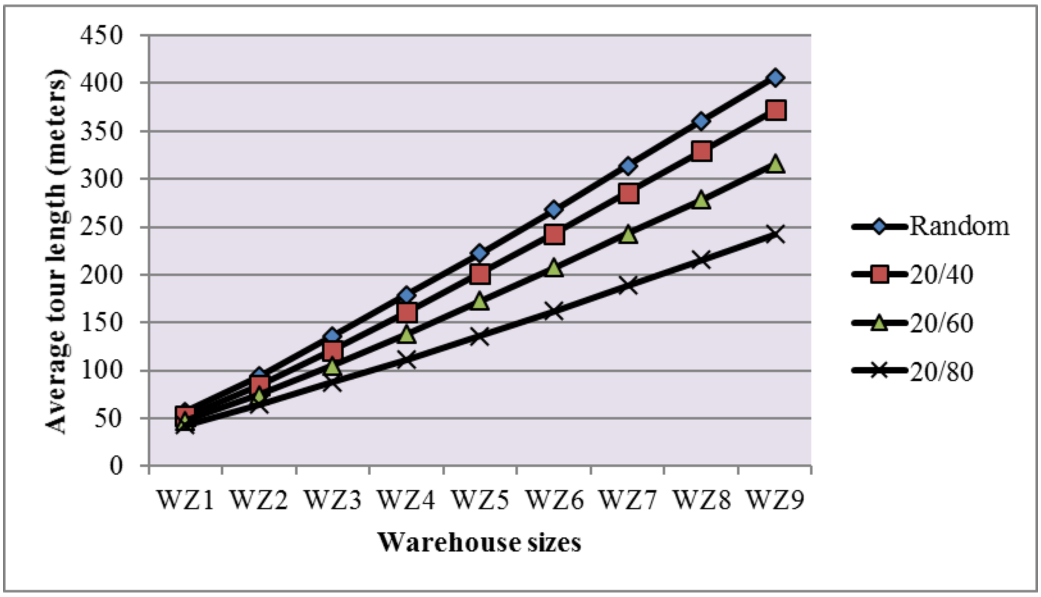


Figure 4.12 Average tour length for the exact algorithm in combination with various storage assignment policies for different warehouse sizes.

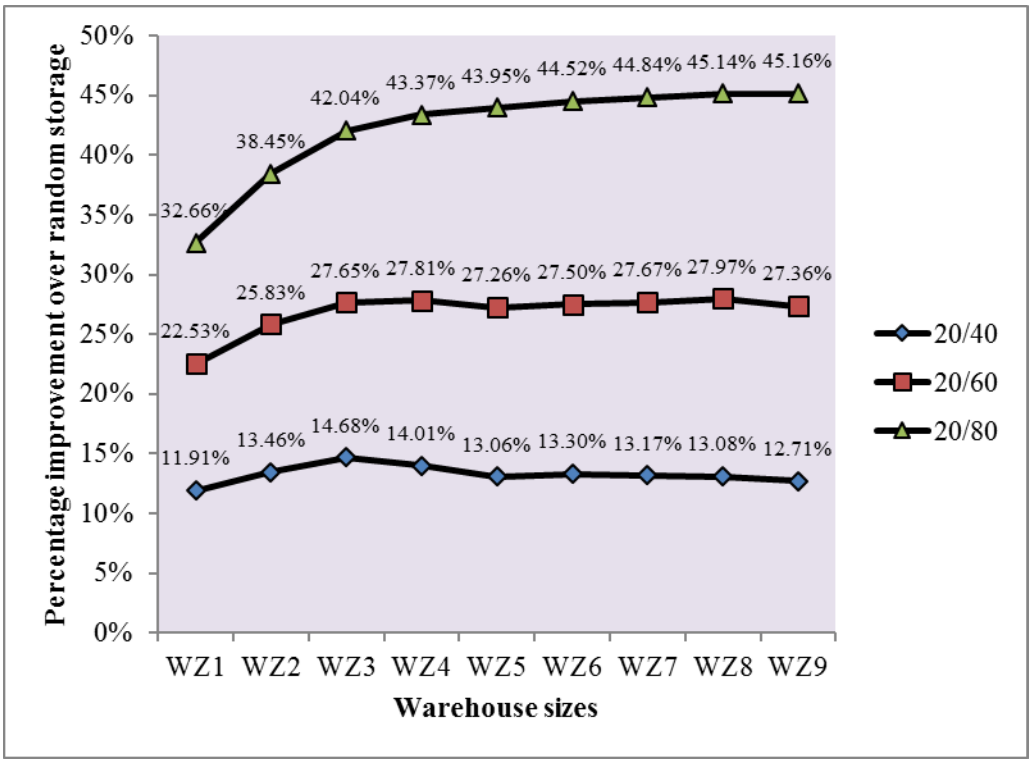


Figure 4.13 Percentage performance advantage of turnover-based storage over random storage for different warehouse sizes.

Table A.3 to Table A.6 (in the appendix) show percentage optimality gaps of the heuristics under the four storage assignments: random storage with uniform demand and turnover-based storage with 20/40, 20/60, and 20/80 demand skewness, respectively. The percentage optimality gaps are calculated as $(Z^h - Z^e)/Z^e$, where Z^h and Z^e represent the average tour lengths resulting from the heuristic (h) and the exact (e) algorithm, respectively (see also Çelik and Süral, 2019). There are a few things to point out from the tables. First, the case of pick-list size 1 is special as all our three heuristics give optimal tours. Secondly, the percentage optimality gap of *CS* increases when the warehouse size increases. This is due to the fact that the order picker has to traverse long picking aisles in large warehouses completely. Furthermore, since the *CL* policy allows the order picker to walk in a picking aisle as far as the largest gap instead of the midpoint, the *CL* policy is an improvement of the *CM* policy. Hence, the average tour length of the *CL* policy is always less than the one resulting from the *CM* policy, which is consistent with the result obtained by Hall (1993). This leads to smaller optimality gaps for the *CL* policy. **Figure 4.14** compares the performance of the three heuristics. The figure implies that the *CS* policy is only beneficial for small warehouses, as the gap continues to increase when the warehouse size increases. The *CM* and *CL* policies, in turn, are advantageous for larger warehouses, as the gap slightly changes when the warehouse size increases.

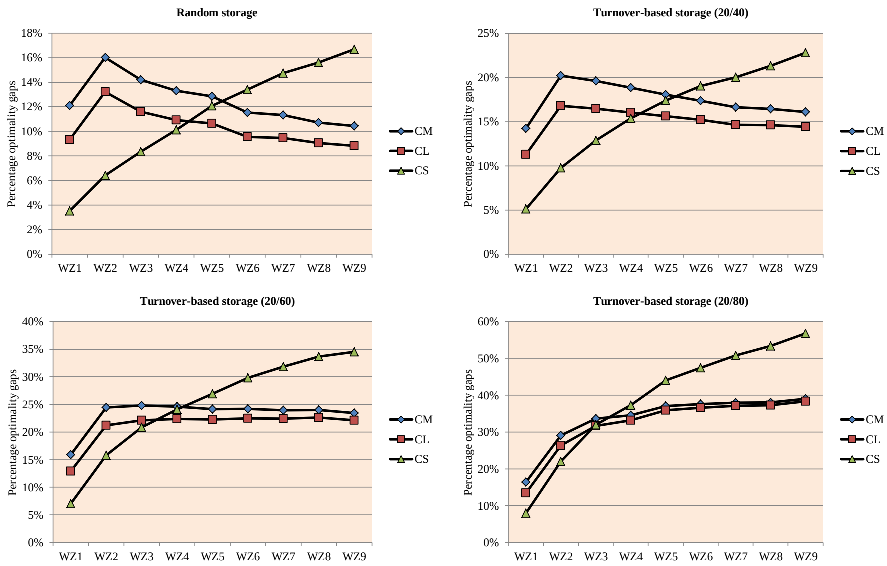


Figure 4.14 Average percentage optimality gaps of the heuristics under different warehouse sizes and storage assignment policies.

The last experiment compares the performance of the chevron warehouse and the conventional two-block warehouse using the average tour length generated by the exact algorithm, under different warehouse sizes, pick-list sizes, and storage assignment policies with different demand skewness. For each instance, the chevron warehouse is compared with a particular two-block warehouse with quadratic blocks and a number of storage locations that is as close as possible to the number of storage locations of the chevron warehouse. The warehouses WZ1-WZ9 are thus compared to two-block warehouses with 2, 4, 5, 7, 8, 10, 11, 13, 14 aisles (in each block) and with 8, 11, 17, 19, 25, 28, 33, 36, 41 storage locations (on each side of an aisle), respectively. The percentage gap is calculated as $\left(Z^{che} - Z^{con}\right)/Z^{con}$, where Z^{che} and Z^{con} represent the average tour lengths for the chevron (*che*) and the conventional two-block warehouses (*con*), respectively (Çelik and Süral, 2014).

The results presented in Table A.7 (in the appendix) and Figure 4.15 lead to the following observations. First, it is clear from Figure 4.15 that the conventional two-block warehouse outperforms the chevron warehouse for all investigated warehouses with average percentage gaps of 24.78%, 24.70%, 25.71%, and 29.50% for the exact algorithms in combination with random storage, and for 20/40, 20/60, and 20/80 demand skewness, respectively. Even though all average gaps are positive, they decrease when the investigated warehouse sizes increase, and seem to become stable at around 11-18%. For large warehouses, the average gap between the chevron warehouse and the conventional two-block warehouse with two square parts are therefore approximately 11-18%. Secondly, Figure 4.15 shows that using a turnover-based storage assignment policy with demand skewness, instead of a random storage assignment policy, has only a small effect on the percentage gap. This is a surprising finding as one might expect a huge impact of turnover-based storage with demand skewness on the chevron warehouse because it has five aisles meeting at the depot (compared to only three aisles in the two-block warehouse), which could make the locations closer to the depot more accessible. However, we should take into account the number of aisles and the effect of empty aisles. Chevron warehouses contain more aisles and have longer distance between aisles. When demand skewness increases, the items in the picking order tend to be closer to the depot. Chevron warehouses still have a greater number of aisles to cover; some aisles rarely contain required items in this case, but these aisles are of shorter length than those located close to the depot. In contrast, two-block warehouses contain a smaller number of aisles of equal length, and hence, benefit more from empty aisles.

Last, Table A.7 shows that there are many instances where the chevron warehouse outperforms the conventional two-block warehouse. We note, however, that these instances are all single-command operations (pick-list size = 1 with only a single required item) in large warehouses. In addition, the performance advantage of the chevron warehouse for single-command operations decreases when demand skewness increases. This emphasizes that a turnover-based assignment under demand skewness may, in some situations, also have a negative effect on the performance of the chevron warehouse, especially in single-command operations.

Figure 4.15 Average percentage gaps between the average tour length for conventional two-block and chevron warehouses.

4.7 Conclusion

This chapter studied the problem of routing an order picker in a manual picker-to-parts order picking system that uses the chevron layout. We proposed an optimal order picker routing policy based on the solution procedures proposed by RR and Roodbergen and de Koster (2001a) that applied graph theory and a dynamic programming procedure. We investigated the effect of different storage assignment policies on the order picking tour obtained by the proposed exact algorithm. Moreover, we modified two routing heuristics, namely the *midpoint* and the *largest gap* heuristics, to make them applicable to the chevron warehouse, referred to as the *chevron midpoint* and *chevron largest gap* heuristics. The average order picking tour lengths resulting from the exact algorithm and the two routing heuristics proposed in this study as well as from the *chevron S-shape* heuristic proposed by Dukic and Opetuk (2012) were compared to evaluate the performance of the routing heuristics. The results indicate that the tours resulting from the exact algorithm are 10.29% to 39.08% shorter than the tours generated by the three routing heuristics. This emphasizes that an optimal order picker routing policy should be the preferred means of guiding the order picker through a chevron warehouse. In this chapter, we also compared the performance of the chevron warehouse to the conventional two-block warehouse by comparing the average tour length generated by the exact algorithm, which was not possible so far as an optimal routing policy had not been proposed for the chevron warehouse yet. Our results imply that conventional two-block warehouses outperform the chevron warehouse especially for large picklists. Given that there is empirical evidence that warehouses following the chevron layout are used in practice, the results obtained in this chapter contribute to improving the efficiency of these warehouses. Based on our findings, companies operating chevron warehouses for regular order picking activities (with no or only a few single-command operations) should evaluate whether switching to a regular two-block layout might be worthwhile. The routing methods proposed in this chapter support such an evaluation.

For future research, it would be interesting to extend the present work to other warehouse layouts, e.g., leaf or butterfly. It would further be worthwhile to extend our work to the case of picker congestion. Furthermore, future research could investigate the case of a dynamic (online) order picking system where a pick-list that is currently in progress can be updated any time. In this chapter, we heuristically modified the turnover-based storage assignment policy; developing an optimal turnover-based storage assignment policy for the chevron warehouse would hence be an interesting extension for future research. Since the performance of an order picker routing policy usually also depends on the order batching strategy, future research could also study the effect of different batching strategies on the performance of the proposed routing policies.

Chapter 5 Order picker routing in the leaf warehouse[7]

5.1 Introduction

As mentioned in Chapter 2, one of the observations from our review is that the order picker routing problem in non-conventional warehouses has not received much attention yet. The non-conventional leaf warehouse has so far only been studied from a layout design perspective using the travel distance model (see Öztürkoğlu et al., 2012). The routing of order pickers in the leaf warehouse has not been studied at all so far. This chapter intends to close this research gap by proposing an exact order picker routing algorithm for the leaf warehouse using the concepts of RR and Roodbergen and de Koster (2001a). Moreover, we propose four simple routing heuristics, referred to as the *leaf S-shape (LS)*, *leaf return (LR)*, *leaf midpoint (LM)*, and *leaf largest gap (LL)* heuristics. To the best of our knowledge, this dissertation is the first study that proposes order picker routing policies for the leaf warehouse. In computational experiments, we evaluate the performance of the heuristics under different demand distributions and storage assignment policies used in warehouses. The remainder of this chapter is organized as follows. Section 5.2 describes the problem investigated in this chapter. Section 5.3 then proposes a procedure for optimally routing order pickers through the leaf warehouse, and Section 5.4 develops corresponding routing heuristics. The results of computational experiments are presented in Section 5.5. Finally, Section 5.6 concludes this chapter and presents an outlook on future research opportunities.

5.2 Problem description and graph representation

This chapter focuses on order picker routing in the leaf warehouse (illustrated in Figure 5.1), a layout that was first described by Öztürkoğlu et al. (2012). As can be seen, the leaf warehouse consists of two horizontal cross aisles, referred to as the front and back cross aisles, two vertical cross aisles, referred to as the left and right vertical cross aisles, and two diagonal cross aisles, referred to as the left and right diagonal cross aisles. The study at hand divides the leaf warehouse into four parts, which will be referred to as parts 1, 2, 3, and 4 in the following (the numbering of the parts is illustrated in Figure 5.1). We assume that parts 1 and 4 consist of the same number of picking aisles, n_s. Similarly, parts 2 and 3 are composed of the same number of picking aisles, n_t, where each picking aisle in both parts consists of two sub-aisles, namely a vertical and a diagonal sub-aisle (this is illustrated in Figure A.17 in the appendix).

[7] Chapter 5 is based on the following working paper: Masae, M., Glock, C.H., Vichitkunakorn, P., A method for efficiently routing order pickers in the leaf warehouse.

We also assume that the front end of a vertical sub-aisle intersects with the front end of a diagonal sub-aisle (see Figure A.17). The leaf warehouse under study has a single depot in the middle of the front cross aisle, where picked items are dropped off or where the order picker receives new pick-lists. Furthermore, we assume that all picking aisles are narrow, such that the order picker can retrieve the requested items from both sides of the picking aisles without facing an additional travel distance for crossing the aisle. We consider the routing problem for a single order picker and ignore possible interdependencies with other order pickers working in the same picking aisle, e.g. picker blocking. We further focus on the order picker routing problem in a low-level picker-to-parts system, such that the requested items can be picked directly from the racks without additional vertical travel. We also assume a single storage system where each item type is stored in a single location only. The order picking process in our case works as follows: The order picker receives a pick-list containing a list of items to be picked at the depot. S/he then walks through the picking area for retrieving the items contained on the pick-list from their storage locations until all requested items have been obtained, and then returns to the depot. The requested items are picked according to the pick-by-order principle, which means the order picker handles a single customer order in each picking tour.

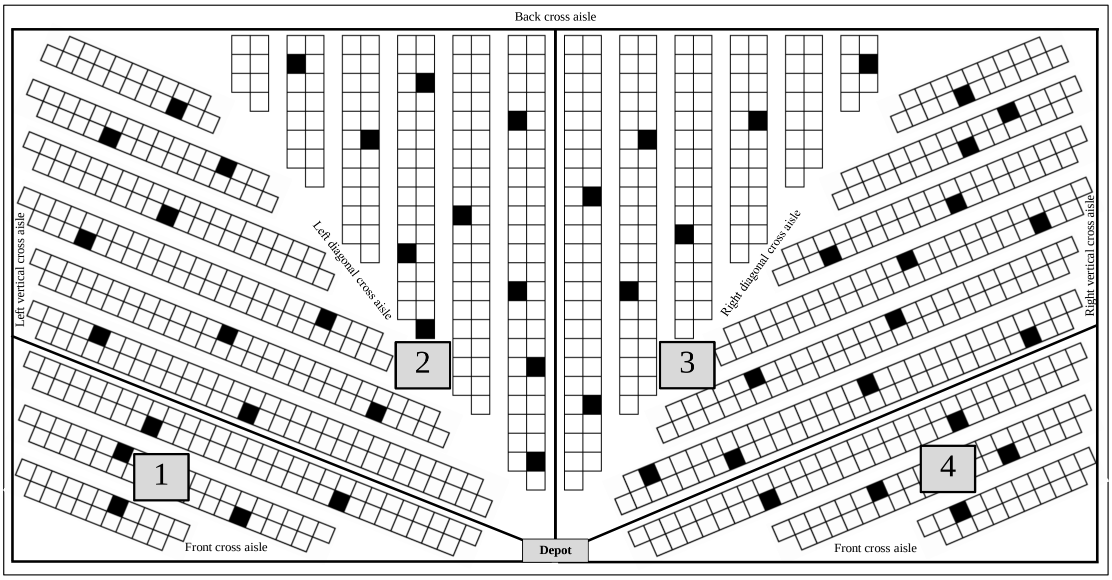


Figure 5.1 Leaf warehouse with 50 requested items (marked with black boxes).

To solve the order picker routing problem in the leaf warehouse, we first define a graph representation G of the warehouse with a set of vertices and edges as shown in **Figure 5.2**. The vertices in our case represent the depot location, the storage locations, and the intersection of each picking aisle with a cross aisle. Any pair of vertices that have a direct path connecting them without passing through other vertices are connected by infinite edges (multi-edges).

Again in this chapter, we use the vertex v_0 to denote the location of the depot, while the vertices $v_i, i = 1, 2, \ldots, m$, represent the storage locations of all m items requested on the pick-list. In part 1 of the leaf warehouse, we assume that the picking aisles are numbered from left to right, where the first picking aisle of part 1 is the left-most aisle from the depot. Conversely, the picking aisles in part 4 are numbered from the right to left. For $j = 1,2, \ldots, n_s$, the vertices $c_{j,1}$ and $b_{j,1}$ represent the intersection of picking aisle j in part 1 with the left vertical and front cross aisles, respectively. The vertices $c_{j,4}$ and $b_{j,4}$ represent the intersection of picking aisle j in part 4 with the right vertical and front cross aisles, respectively. As to parts 2 and 3, the picking aisles are numbered from the back to the front, where the first picking aisle of parts 2 and 3 is the back-most aisle from the depot. For $j = 1,2, \ldots, n_t$, the vertices $a_{j,2}$, $b_{j,2}$, and $c_{j,2}$ represent the intersection of each picking aisle j in part 2 with the back, the left diagonal, and the left vertical cross aisles, respectively. The intersection of each picking aisle j in part 3 with the back, the right diagonal, and the right vertical cross aisles are denoted by the vertices $a_{j,3}$, $b_{j,3}$, and $c_{j,3}$, respectively. We note that some vertices coincide, including (1) $c_{n_s,1} = c_{n_t,2}$, (2) $c_{n_s,4} = c_{n_t,3}$, (3) $b_{n_s,1} = b_{n_t,2} = b_{n_t,3} = b_{n_s,4}$, and (4) $a_{n_t,2} = a_{n_t,3}$. We emphasize the three aisles printed in bold in **Figure 5.2** that represent the two diagonal cross aisles and the vertical middle aisle. These aisles connect the different parts of the warehouse, and therefore they are referred to as the connection aisles in the following. The vertices along these connection aisles as well as the coincident vertices mentioned above have been highlighted using dotted circles. Such vertices require a special treatment in our solution procedure. **Figure 5.2** shows the graph representation G associated with the warehouse instance in **Figure 5.1**, where $n_s = 4$, $n_t = 7$, and $m = 50$. For simplicity, the vertex v_0 is added between $b_{3,1}$ and $b_{4,1}$, instead of setting it equal to $b_{4,1}$.

5.3 Exact routing algorithm for the leaf warehouse

As mentioned earlier, the order picker routing problem in the leaf warehouse is identical to the problem of finding the shortest picking tour in its graph representation. To find the shortest picking tour in G, we first construct the minimum-length tour subgraph on G using a dynamic programming procedure. Secondly, we generate the optimal order picking tour from the minimum-length tour subgraph. Section 5.3.1 introduces the basic definitions used for constructing the minimum-length tour subgraph. Section 5.3.2 then describes the procedure for constructing the minimum-length tour subgraph. The tour construction algorithm is finally presented in Section 5.3.3.

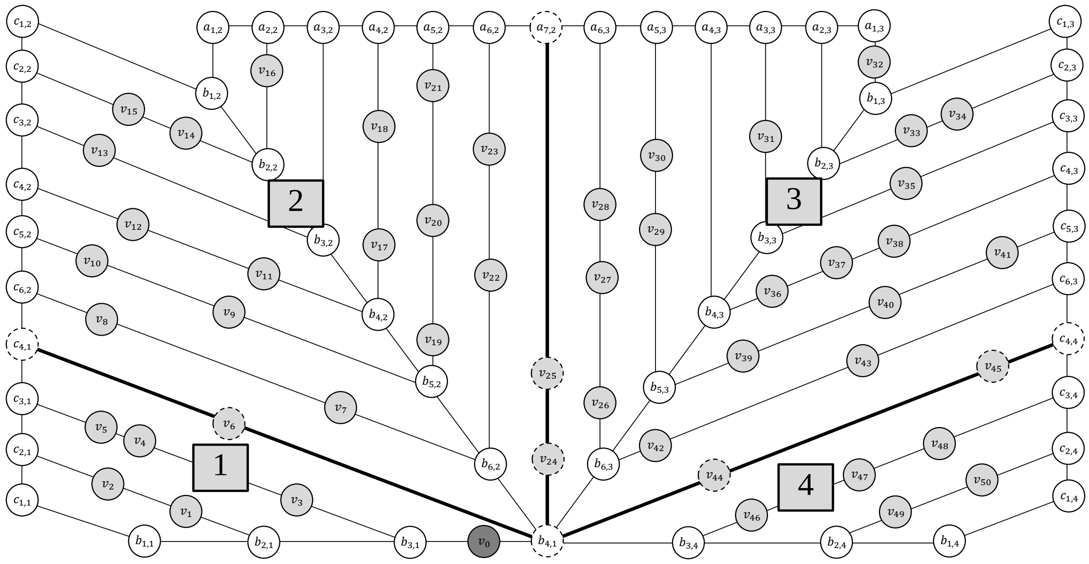


Figure 5.2 Graph representation G of a leaf warehouse with $n_s = 4$, $n_t = 7$, and $m = 50$.

5.3.1 Basic definitions

The following definitions are used for constructing the minimum-length tour subgraph.

Definition 5.1 Let G be the graph representation with n_s, n_t, n_t, and n_s picking aisles in parts 1, 2, 3, and 4 of the warehouse (as defined in Figure 5.2), respectively. For $j \in \{1, 2, \ldots, n_s\}$, we let

- $L_{j,1}^{-}$ be the subgraph (in part 1) of G consisting of the vertices $c_{j,1}$ and $b_{j,1}$ together with all vertices and edges in/connected to the aisles in part 1 of index lower than j,
- $L_{j,1}^{+}$ be the subgraph of G consisting of $L_{j,1}^{-}$ and all vertices and edges between the vertices $c_{j,1}$ and $b_{j,1}$,
- $L_{j,4}^{-}$ be the subgraph (in part 4) of G consisting of the vertices $c_{j,4}$ and $b_{j,4}$ together with all vertices and edges in/connected to the aisles in part 4 of index lower than j,
- $L_{j,4}^{+}$ be the subgraph of G consisting of $L_{j,4}^{-}$ and all vertices and edges between the vertices $c_{j,4}$ and $b_{j,4}$.

For $j \in \{1,2, \ldots, n_t\}$, we let

- $L_{j,2}^{-}$ be the subgraph (in part 2) of G consisting of the vertices $a_{j,2}$, $b_{j,2}$, and $c_{j,2}$ together with all vertices and edges in/connected to the aisles in part 2 of index lower than j,
- $L_{j,2}^{+l}$ be the subgraph of G consisting of $L_{j,2}^{-}$ and all vertices and edges between the vertices $b_{j,2}$ and $c_{j,2}$,
- $L_{j,2}^{+u}$ be the subgraph of G consisting of $L_{j,2}^{+l}$ and all vertices and edges between the vertices $a_{j,2}$ and $b_{j,2}$,

- $L_{j,3}^{-}$ be the subgraph (in part 3) of G consisting of the vertices $a_{j,3}$, $b_{j,3}$, and $c_{j,3}$ together with all vertices and edges in/connected to the aisles in part 3 of index lower than j,
- $L_{j,3}^{+l}$ be the subgraph of G consisting of $L_{j,3}^{-}$ and all vertices and edges between the vertices $b_{j,3}$ and $c_{j,3}$,
- $L_{j,3}^{+u}$ be the subgraph of G consisting of $L_{j,3}^{+l}$ and all vertices and edges between the vertices $a_{j,3}$ and $b_{j,3}$,

To simplify notation, we use the notation L_j to indicate that a result holds if we let $L_j = L_{j,1}^{-}$, $L_j = L_{j,1}^{+}$, $L_j = L_{j,4}^{-}$, or $L_j = L_{j,4}^{+}$ for $j \in \{1, 2, \dots, n_s\}$, and $L_j = L_{j,2}^{-}$, $L_j = L_{j,2}^{+l}$, $L_j = L_{j,2}^{+u}$, $L_j = L_{j,3}^{-}$, $L_j = L_{j,3}^{+l}$, $L_j = L_{j,3}^{+u}$ for $j \in \{1,2, \dots, n_t\}$.

Definition 5.2 A picking tour in G is a directed walk that starts from v_0, passes through the vertices $v_i, i = 1, 2, \dots, m$, and ends at v_0, where each edge in G is traversed at most once. A subgraph T of G is a tour subgraph if it is the underlying graph of some picking tour in G.

Theorem 5.1 (Ratliff and Rosenthal, 1983) A subgraph T of G is a tour subgraph if and only if all the following conditions hold.

1. T is connected.
2. The degree of vertices v_i are positive in T for $i = 0, 1, 2, \dots, m$.
3. The degree of any vertex of T is even (possibly zero) in T.

Definition 5.3 (Ratliff and Rosenthal, 1983) For any subgraph L_j of G, a subgraph T_j of L_j is an L_j PTS if there exists a subgraph C of G consisting of edges that are contained in G, but not in L_j, such that $T_j \cup C$ is a tour subgraph of G. The subgraph C is called a completion of the subgraph T_j.

Theorem 5.2 (Ratliff and Rosenthal, 1983) Let L_j be one of the subgraphs of G in part 1 or 4 according to Definition 5.1. Then a subgraph T_j of L_j is an L_j PTS if all the following conditions hold.

1. The degrees of all vertices v_i of L_j, representing item locations, are positive in T_j.
2. Every vertex in T_j, except possibly for a_j and b_j, has even or zero degree.
3. Excluding vertices of zero degree, T_j has either
   - no connected component,
   - a single connected component containing at least one of a_j and b_j,

- two connected components, with each component containing at least one of a_j and b_j.

Theorem 5.3 (Roodbergen and de Koster, 2001a) Let L_j be one of the subgraphs of G in part 2 or 3 according to Definition 5.1. Then a subgraph T_j of L_j is an L_j PTS if all the following conditions hold.

1. The degrees of all vertices v_i of L_j, representing item locations, are positive in T_j.
2. Every vertex in T_j, except possibly for a_j, b_j, and c_j, has even or zero degree.
3. Excluding vertices of zero degree, T_j has either
   - no connected component,
   - a single connected component containing at least one of a_j, b_j, and c_j,
   - two connected components, with each component containing at least one of a_j, b_j, and c_j,
   - three connected components, with each component containing at least one of a_j, b_j, and c_j.

Definition 5.4 (Ratliff and Rosenthal, 1983) Two L_j PTSs, namely T_j^1 and T_j^2, are said to be equivalent if for any completion C_j of T_j^1 such that $T_j^1 \cup C_j$ is a tour subgraph, $T_j^2 \cup C_j$ is also a tour subgraph, and vice versa. In other words, the set of completions of T_j^1 and T_j^2 coincide.

Theorem 5.4 Let L_j be one of the subgraphs of G in part 1 or 4 according to Definition 5.1. Two L_j PTSs T_j^1 and T_j^2 are equivalent if and only if all the following conditions hold.

1. The degrees parity of the vertex b_j (resp. c_j) in both T_j^1 and T_j^2 are equal.
2. The vertices b_j and c_j are connected in T_j^1 if and only if they are connected in T_j^2.

This result and its proof coincide with the result for the single-block conventional warehouse shown in RR, and we will use their notation for equivalence classes of L_j PTSs in part 1 or 4 of G.

Theorem 5.5 Let L_j be one of the subgraphs of G in part 2 or 3 according to Definition 5.1. Two L_j PTSs T_j^1 and T_j^2 are equivalent if and only if all the following conditions hold.

1. The degrees parity of the vertex a_j (resp. b_j and c_j) in both T_j^1 and T_j^2 are equal.

2. The pair of vertices a_j and b_j (resp. pair of b_j and c_j, and pair of c_j and a_j) are connected in T_j^1 if and only if they are connected in T_j^2.

This result again coincides with the result for the two-block conventional warehouse shown in Roodbergen and de Koster (2001a), and their notation for equivalence classes of L_j PTSs will be used for equivalent classes of L_j PTSs in part 2 or 3 of G.

5.3.2 Constructing the minimum-length tour subgraph

The solution procedure for finding the minimum-length tour subgraph of the entire graph G consists of three main steps (see Sections 5.3.2.1 to 5.3.2.3). In the first step, the algorithm of RR is used to construct the minimum-length $L_{n_s,1}^+$ and $L_{n_s,4}^+$ PTSs in parts 1 and 4, respectively. In the second step, the algorithm of Roodbergen and de Koster (2001a) is applied to construct the minimum-length $L_{n_t,2}^{+u}$ and $L_{n_t,3}^{+u}$ PTSs in parts 2 and 3, respectively. In the last step, the minimum-length L_j PTSs resulting from the previous steps are combined to connect the different parts of the warehouse. Theorem 5.6 guarantees that the combined subgraph is a tour subgraph. We assume that the vertices corresponding to the storage locations along the left and right diagonal cross aisles are contained in parts 1 and 4, respectively. The vertices corresponding to the storage locations along the vertical middle aisle are contained in part 2. Therefore, there are no vertices along the last aisle n_t of part 3 as shown in Figure 5.3(c).

5.3.2.1 Constructing the minimum-length $L_{n_s,1}^+$ and $L_{n_s,4}^+$ PTSs

The equivalence classes of L_j PTSs in part 1 are referred to by the triple (degree parity of $c_{j,1}$, degree parity of $b_{j,1}$, number of connected components). In part 4, the equivalence classes of L_j PTSs are represented by (degree parity of $c_{j,4}$, degree parity of $b_{j,4}$, number of connected components). Similar to RR, we apply a dynamic programming procedure to find the minimum-length tour subgraph by defining the states of the algorithm as the equivalence classes of L_j PTSs. The transitions between states are the addition of vertical or horizontal components to PTSs, and the cost in each transition is the sum of edge weights of an L_j PTS. We first add vertical components from **Figure A.1** to the $L_{1,1}^-$ PTS to construct $L_{1,1}^+$ PTSs. The minimum-length $L_{1,1}^+$ PTSs in each equivalence class are extended to generate $L_{2,1}^-$ PTSs by adding horizontal components from **Figure A.1** to the $L_{1,1}^+$ PTSs. We further construct $L_{2,1}^+$ PTSs by adding vertical components from **Figure A.1** to the minimum-length $L_{2,1}^-$ PTSs in each equivalence class. Following the same procedure for the aisles $j = 3$ to $j = n_s$ in part 1, all $L_{n_s,1}^+$ PTSs have been obtained in the last step. The same procedure is applied to part 4 to obtain the minimum-length $L^{+}_{n_s,4}$ PTSs in each equivalence class. Figure 5.3(a) and Figure 5.3(d) show instances of $L^{+}_{4,1}$ and $L^{+}_{4,4}$ PTSs of G from Figure 5.2.

5.3.2.2 Constructing the minimum-length $L^{+u}_{n_t,2}$ and $L^{+u}_{n_t,3}$ PTSs

The equivalence classes of all L_j PTSs, $j \in \{1,2,\dots,n_t\}$, in parts 2 and 3 are denoted by (degree parity of $a_{j,2}$, degree parity of $b_{j,2}$, degree parity of $c_{j,2}$, number of connected components, connectivity) and (degree parity of $a_{j,3}$, degree parity of $b_{j,3}$, degree parity of $c_{j,3}$, number of connected components, connectivity), respectively. We construct the minimum-length $L^{+u}_{n_t,2}$ and $L^{+u}_{n_t,3}$ PTSs of each equivalence class by considering each aisle $j \in \{1,2,\dots,n_t\}$ in sequence. First, we add each vertical component from **Figure A.1** to the $L^{-}_{1,2}$ PTS between $b_{1,2}$ and $c_{1,2}$, resulting in $L^{+l}_{1,2}$ PTSs. The minimum-length $L^{+l}_{1,2}$ PTSs in each equivalence class are extended by adding each vertical component from **Figure A.1** to the $L^{+l}_{1,2}$ PTSs in-between $a_{1,2}$ and $b_{1,2}$ to obtain $L^{+u}_{1,2}$ PTSs. Using only the minimum-length $L^{+u}_{1,2}$ PTSs in each equivalence class, $L^{-}_{2,2}$ PTSs are constructed by adding each horizontal component from Figure 3.6 (The reader is referred to Chapter 3) to the $L^{+u}_{1,2}$ PTSs. The minimum-length $L^{-}_{2,2}$ PTSs in each equivalence class are extended by adding vertical components in the lower and upper blocks, respectively. The algorithm continues in this fashion for the next aisles $j = 3$ to $j = n_t$ until we obtain all $L^{+u}_{n_t,2}$ PTSs. Following the same procedure for part 3, we obtain the minimum-length $L^{+u}_{n_t,3}$ PTSs in each equivalence class. Figure 5.3(b) and Figure 5.3(c) show examples of $L^{+u}_{7,2}$ and $L^{+u}_{7,3}$ PTSs of G from Figure 5.2.

5.3.2.3 Combining the minimum-length tour subgraphs

We connect parts 1 and 2 of G at the connection aisle by combining the minimum-length $L^{+}_{n_s,1}$ PTSs in each equivalence class with the minimum-length $L^{+u}_{n_t,2}$ PTSs in each equivalence class, following the transition Table A.8 in the Appendix. The resulting PTSs from this combination are referred to as $L^{12}_{n_t}$ in the following. After that, we connect parts 3 and 4 by combining the minimum-length $L^{+u}_{n_t,3}$ PTSs with the minimum-length $L^{+}_{n_s,4}$ PTSs, again using the transition Table A.8. The resulting PTSs are referred to as $L^{34}_{n_t}$. We further combine the minimum-length $L^{12}_{n_t}$ PTSs and the minimum-length $L^{34}_{n_t}$ PTSs using the transition Table A.9. The equivalence classes of the resulting PTSs are denoted by (degree parity of $deg(a_{n_t,2}) + deg(a_{n_t,3})$, degree parity of $deg(b_{n_t,2}) + deg(b_{n_t,3})$, degree parity of $c_{n_t,2}$, degree parity of $c_{n_t,3}$, number of connected components, connectivity). Note that the number of possible equivalence classes of each $L^{12}_{n_t}$ and $L^{34}_{n_t}$ is 25. However, these 25 equivalence classes contain some classes that lead to infeasible solutions. For example, an equivalence class $(E, 0, E, 2C)$

of $L_{n_t}^{12}$ never gives a feasible solution as it has two components which cannot be combined into a single component, no matter which equivalence class of $L_{n_t}^{34}$ we select. After deleting all equivalence classes that lead to infeasible solutions, we obtain 14 possible equivalence classes of each $L_{n_t}^{12}$ and $L_{n_t}^{34}$, as shown in Table A.9. Note that only the entries below the diagonal are shown as the table is symmetric along the diagonal. In the end, the shortest PTS from the set of PTSs that are connected and possess even degree parity in the connection vertices, ($(E, E, E, E, 1C)$, $(E, E, E, 0{,}1C)$, $(E, E, 0, E, 1C)$, $(E, E, 0{,}0{,}1C)$, $(E, 0, E, E, 1C)$, $(E, 0, E, 0{,}1C)$, $(E, 0{,}0, E, 1C)$, $(E, 0{,}0{,}0{,}1C)$, $(0, E, E, E, 1C)$, $(0, E, E, 0{,}1C)$, $(0, E, 0, E, 1C)$, $(0, E, 0{,}0{,}1C)$, $(0{,}0, E, E, 1C)$, $(0{,}0, E, 0{,}1C)$, $(0{,}0{,}0, E, 1C)$ and $(0{,}0{,}0{,}0{,}1C)$), is selected as the minimum-length tour subgraph of the whole graph G.

Theorem 5.6 T is a tour subgraph of G if and only if T decomposes into four subgraphs: $L_{n_s,1}^{+}$ PTS (T_1), $L_{n_t,2}^{+u}$ PTS (T_2), $L_{n_t,3}^{+u}$ PTS (T_3), and $L_{n_s,4}^{+}$ PTS (T_4), which all satisfy the following five conditions.

1. $T_1 \cup T_2 \cup T_3 \cup T_4$ is connected.
2. $\deg_{T_2}(a_{n_t,2}) + \deg_{T_3}(a_{n_t,3})$ is even or zero.
3. $\deg_{T_1}(b_{n_s,1}) + \deg_{T_2}(b_{n_t,2}) + \deg_{T_3}(b_{n_t,3}) + \deg_{T_4}(b_{n_s,4})$ is even or zero.
4. $\deg_{T_1}(c_{n_s,1}) + \deg_{T_2}(c_{n_t,2})$ is even or zero.
5. $\deg_{T_3}(c_{n_t,3}) + \deg_{T_4}(c_{n_s,4})$ is even or zero.

Proof. To prove sufficiency, we assume that T decomposes into four subgraphs T_1, T_2, T_3, and T_4 satisfying the conditions (1)-(5). Since T_1, T_2, T_3, and T_4 are $L_{n_s,1}^{+}$, $L_{n_t,2}^{+u}$, $L_{n_t,3}^{+u}$, and $L_{n_s,4}^{+}$ PTSs, respectively, the degrees of all vertices v_i are positive in T. The conditions (2)-(5) imply that the degree of the other vertices of T is even or zero. Moreover, the condition (1) implies that T is connected. Hence, T is a tour subgraph of G. To prove necessity, we assume that T is a tour subgraph of G. By Definition 5.2, T is the underlying graph of a picking tour in G that starts from v_0, passes through the vertices $v_i, i = 1,2,\ldots,m$, and ends at v_0, where each edge in G is traversed at most once. By Theorem 5.1, the degree of the v_i is even and positive, while the degree of the other vertices is even or zero. We define four subgraphs: $T_1 = T \cap L_{n_s,1}^{+}$, $T_2 = T \cap L_{n_t,2}^{+u}$, $T_3 = T \cap L_{n_t,3}^{+u}$ and $T_4 = T \cap L_{n_s,4}^{+}$. It is easy to see that T_1, T_2, T_3 and T_4 are $L_{n_s,1}^{+}$ PTS, $L_{n_t,2}^{+u}$ PTS, $L_{n_t,3}^{+u}$ PTS, and $L_{n_s,4}^{+}$ PTS, respectively. In addition, the conditions (1)-(5) hold. This proves necessity.

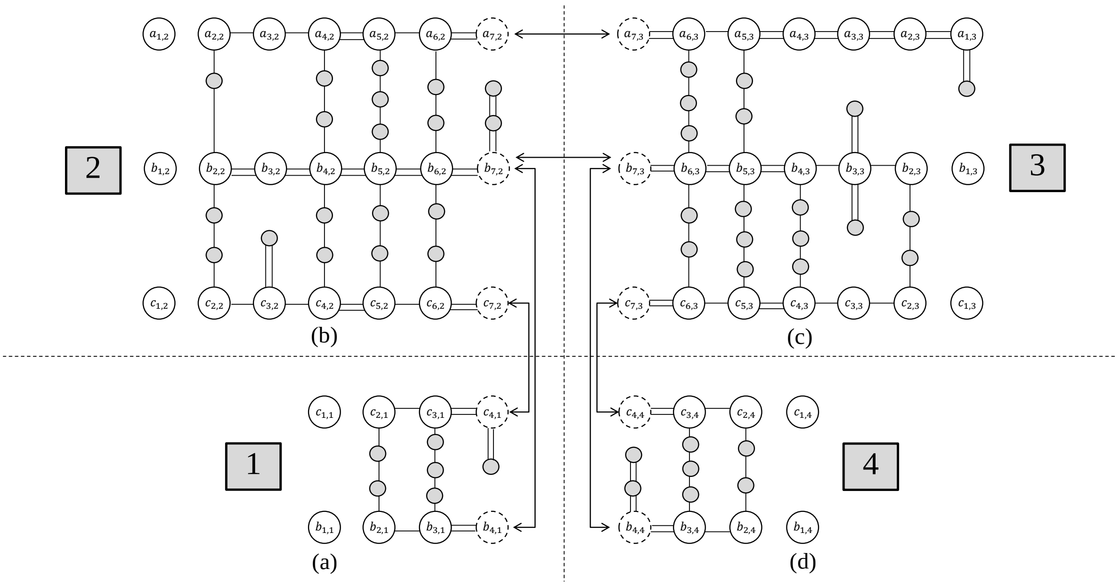


Figure 5.3 An L_j PTS in each part of G form **Figure 5.2**.

5.3.3 Algorithm for constructing the picking tour

This section presents the procedure for constructing the minimum-length picking tour in the leaf warehouse from the minimum-length tour subgraph of its graph representation. We use the picking tour construction algorithm presented in RR to our case. The algorithm can be summarized as follows:

Step 1. Start the picking tour at the vertex v_0 as the first vertex visited.

Step 2. Let v^* be the vertex that is currently being visited.

Step 3. If there is a pair of unused parallel edges incident to v^*, select one of them to move to the next vertex, then go back to *Step 2*. Otherwise, continue to the next step.

Step 4. If there are any unused single edges that are not among the pairs of parallel edges from *Step 3* incident to v^*, select one of them to move to the next vertex, then go back to *Step 2*. Otherwise, continue to the next step.

Step 5. If there is a pair of parallel edges incident to v^* including one unused edge, select it to move to the next vertex, then go back to *Step 2*. Otherwise, continue to the next step.

Step 6. The picking tour ends at the vertex v_0 as the last vertex visited.

5.4 Simple routing heuristics for the leaf warehouse

The use of routing heuristics is dominant in practice (Petersen, 1999; Petersen and Aase, 2004; Henn, 2012). Their importance in industry is reflected by the high attention routing heuristics received in the scientific literature in the past. In the following, we propose alternative routing heuristics for the leaf warehouse. Çelik and Süral (2014) and Masae et al. (2019)

adapted various routing heuristics that had been proposed earlier for conventional warehouses to the fishbone and chevron warehouses, respectively. Their heuristics treat each part of the fishbone or chevron warehouse as a single-block warehouse. Motivated by their ideas, we divide the leaf warehouse into three parts, referred to as the left, the middle, and the right parts, and we treat each part as a single-block warehouse. Çelik and Süral (2014) developed the *fishbone S-shape* and *fishbone largest gap* routing heuristics. We modify these two routing heuristics to make them applicable to the leaf warehouse, and we refer to them as the *leaf S-shape* and *leaf largest gap* heuristics. Moreover, we propose two additional routing heuristics, referred to as the *leaf return* and the *leaf midpoint* heuristics. The four routing heuristics can be summarized as follows:

Leaf S-shape (LS): The order picker starts at the depot and moves to the left-most aisle of the left part that contains at least one requested item. S/he traverses this aisle completely and moves to the next aisle (in the same part) containing a requested item, and continues according to the same procedure until all requested items in the left part have been retrieved. S/he then moves to the left-most aisle of the middle part containing requested items and starts retrieving all requested items according to the same procedure until the right-most aisle of the middle part that contains at least one requested item has been completed. Afterwards, s/he continues picking all requested items stored in the right part from the back- to the front-most aisle containing requested items and then returns to the depot. Figure 5.4 presents the solution of the *leaf S-shape* heuristic for the example given in Figure 5.1.

Leaf return (LR): The order picker starts at the depot, moves to the left-most aisle of the left part that contains at least one requested item, and picks the requested items by entering and leaving the aisle from the front end. The same procedure is used for the next aisles containing requested items in the left part until this part has been completed. The order picker repeats the same picking process in the middle and right parts of the leaf warehouse and finally returns to the depot. Figure 5.5 illustrates the route created by the *leaf return* heuristic for the example introduced earlier.

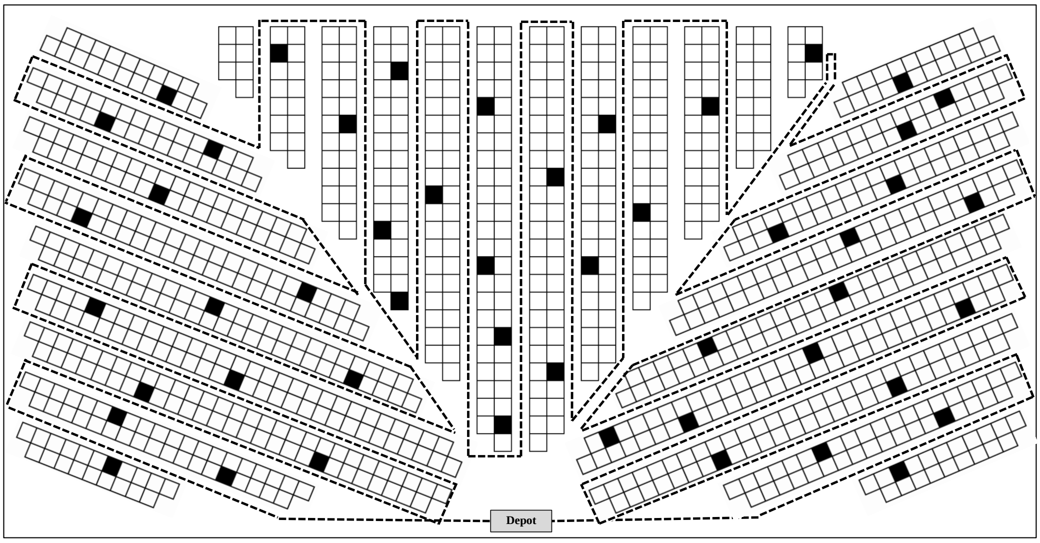


Figure 5.4 The route resulting from the *leaf S-shape* heuristic for the example from **Figure 5.1**.

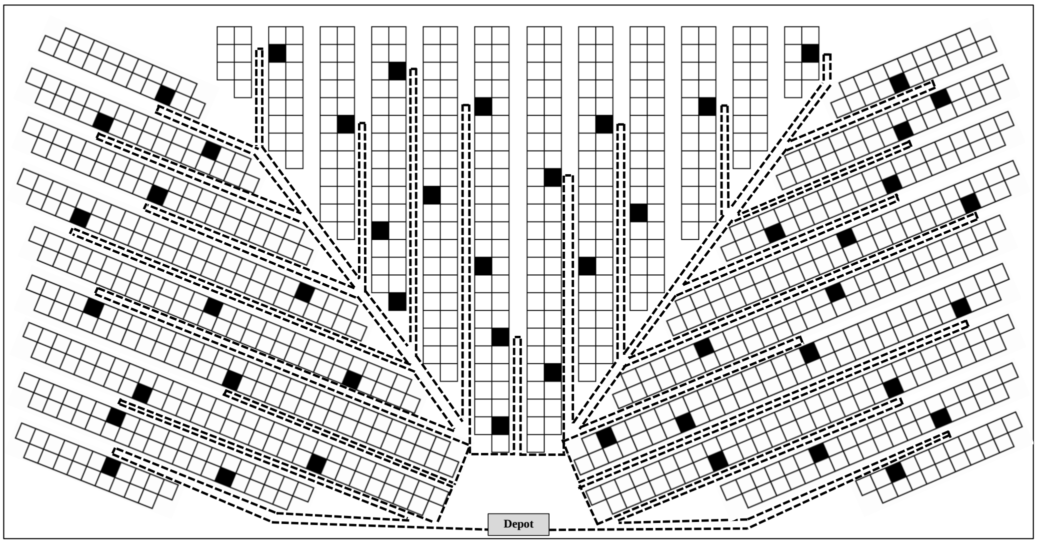


Figure 5.5 The route resulting from the *leaf return* heuristic for the example from Figure 5.1.

Leaf midpoint (LM): This heuristic divides each aisle in each part of the leaf warehouse into two equal halves, referred to as the upper and the lower sections. Starting from the depot, the order picker moves to the left diagonal cross aisle and retrieves the requested items stored in the lower sections of the left and middle parts of the leaf warehouse. S/he then moves to the back cross aisle by traversing the left-most aisle of the middle part that contains at least one requested item and then starts picking the remaining items stored in the upper sections of the middle part. S/he then moves to the right diagonal cross aisle and retrieves the remaining items in the middle part as well as items stored in the lower sections of the right part. After that, the order picker moves to the front cross aisle, the right vertical cross aisle, the back cross aisle, the left vertical cross aisle, and the front cross aisle, respectively, and picks all remaining items stored in the right and left parts. **Figure 5.6** shows the route resulting from the *leaf midpoint* heuristic for the example given in Figure 5.1.

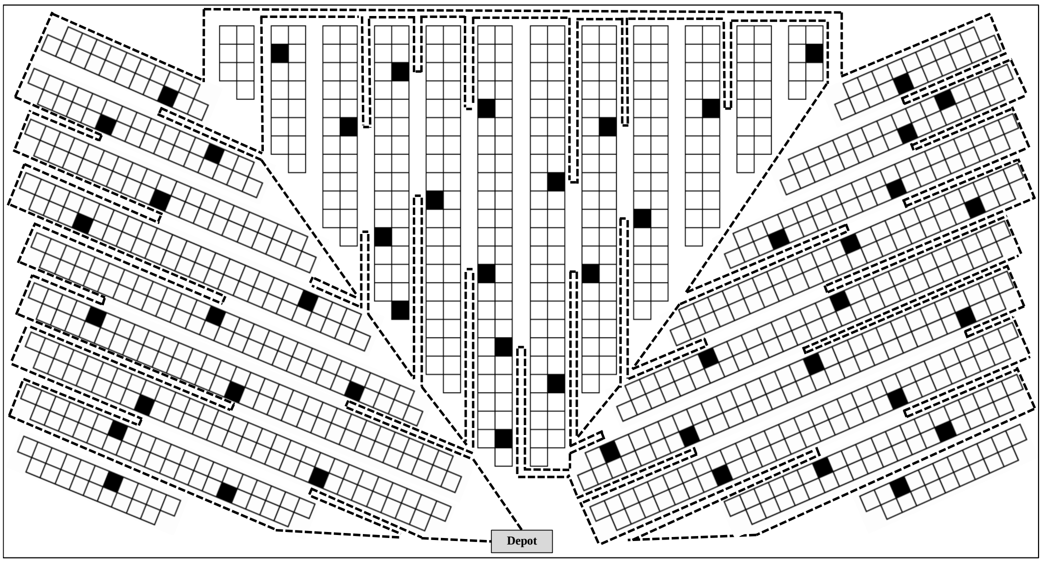


Figure 5.6 The route resulting from the *leaf midpoint* heuristic for the example from Figure 5.1.

Leaf largest gap (LL): This heuristic also divides each aisle in each part of the leaf warehouse into an upper and a lower section according to the largest gap that occurs within an aisle. The largest gap is defined as the longest distance between two requested items in the aisle or between the aisle exits and a requested item. The order picker uses the same procedure than for the *leaf midpoint* heuristic to retrieve all requested items sequentially from the left to the right parts of the leaf warehouse. Figure 5.7 shows the route resulting from the *leaf largest gap* heuristic for the example given in Figure 5.1.

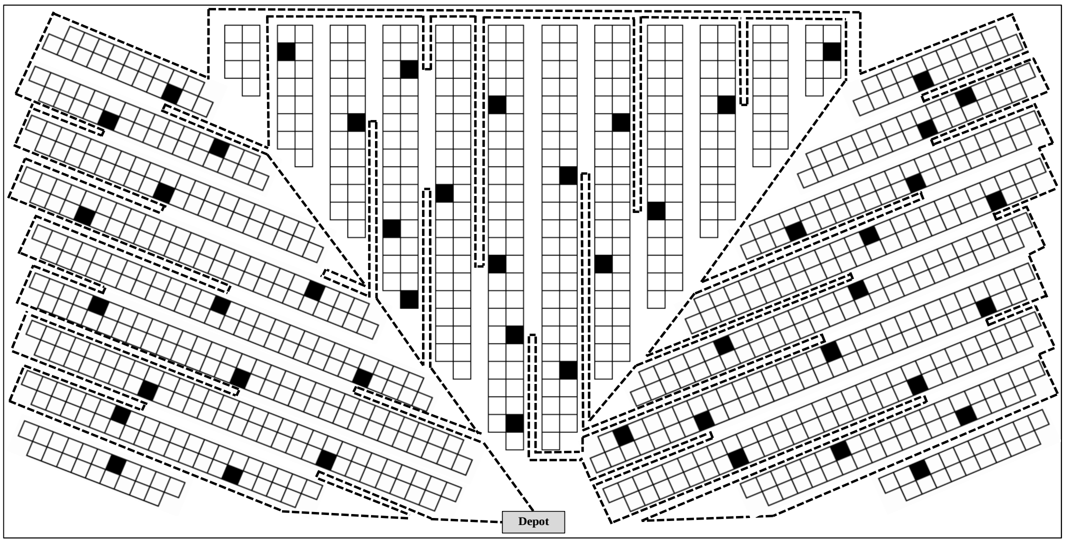


Figure 5.7 The route resulting from the *leaf largest gap* heuristic for the example from **Figure 5.1**.

5.5 Computational experiments

To examine the performance of the exact algorithm and to compare it to the simple routing heuristics proposed in this chapter, we conduct numerical experiments in this section. We vary different model parameters, namely the storage assignment, the size of the leaf warehouse (based on total number of storage locations), and the size of the pick-list, with some problem sets taken from Masae et al. (2019). Four storage assignment policies are considered in our experiment, namely (i) random storage with uniform demand, (ii) turnover-based storage with 20/40 demand skewness, (iii) turnover-based storage with 20/60 demand skewness, and (iv) turnover-based storage with 20/80 demand skewness. For a random storage policy, items are assigned randomly to locations available in the warehouse. In case of turnover-based storage, we used the turnover-based storage with demand skewness presented in Chapter 4. With respect to the size of the leaf warehouse, we consider nine different sizes with a total of 61, 189, 392, 658, 998, 1410, 1880, 2430, and 3050 storage locations, respectively, referred to as WZ1, WZ2, WZ3, WZ4, WZ5, WZ6, WZ7, WZ8, and WZ9. We also consider seven different pick-list sizes with 1, 2, 3, 5, 10, 30, and 60 items. For each of the 63 settings, we randomly generate 1,000 orders to evaluate the average order picking tour length of all routing policies. All routing procedures were implemented in Java, and all instances were run on a computer with Intel Core i5-7200U 2.50 GHz and 8 GB RAM.

To evaluate the performance of the routing heuristics proposed in this chapter, we calculate the percentage optimality gaps using the formula $\left(Z^h - Z^e\right)/Z^e$, where Z^h and Z^e represent the average tour lengths resulting from the heuristic (h) and the exact (e) algorithm, respectively. Table A. 10 (in the appendix) shows the percentage optimality gaps of the four heuristics under the random storage policy. We can see that warehouse size and pick-list size have an impact on the optimality gap of the heuristics. Pick-lists that contain only a single requested item are special cases, as the picking tours obtained by all four heuristics are optimal. With respect to the *LR* policy, the optimality gap increases as the pick-list size increases. The reason is that when the pick density per picking aisle increases, the picker tends to travel from one end of an aisle to the deepest item in the aisle, but still has to return to the end of the aisle s/he entered. This results in a travel distance that is almost twice the aisle length. In terms of the *LS* policy, the optimality gap increases when the pick-list size increases until it reaches the highest percentage optimality gap. After that, the gap gets smaller as the pick-list size increases because the picker almost always traverses every aisle when the pick density is high.

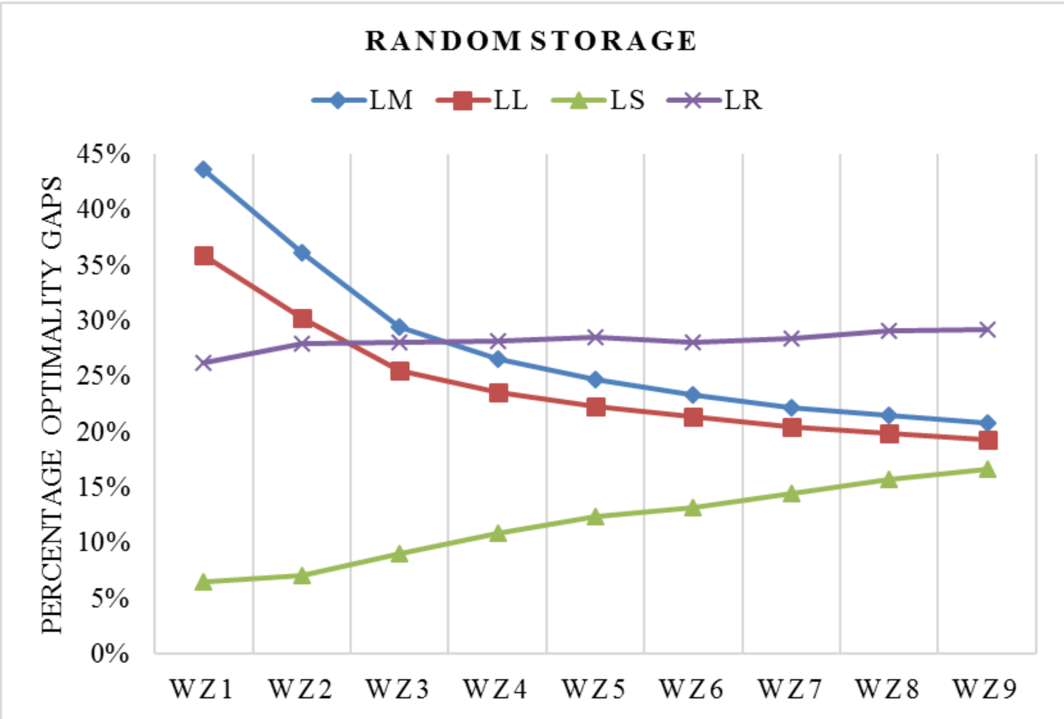


Figure 5.8 Average percentage optimality gaps of the heuristics under different warehouse sizes using random storage.

Figure 5.8 compares the optimality gaps of the four routing heuristics when random storage assignment is applied. As can be seen, *LS* outperforms *LM*, *LL*, and *LR* for all considered warehouses. However, its percentage optimality gap increases in the warehouse size. The larger the warehouse, the more likely it is that the order picker has to completely traverse long picking aisles. The *LM* and *LL* policies have a poor performance in small warehouses, but they perform better when the warehouse size increases. The average percentage optimality gap of the *LL* policy is always less than that of the *LM* policy since the *LL* policy allows the order picker to walk in a picking aisle as far as the largest gap instead of the midpoint, which gives the order picker more flexibility to reduce travel distances. The results thus imply that the *LS* policy is beneficial for all considered warehouses. However, for leaf warehouses with more than 3,050 storage locations, the *LM* and *LL* policies turned out to be good routing policies as well, as their relative performance disadvantages compared to the *LS* policy decreased.

Table A.11 to Table A.13 (in the appendix) show the percentage optimality gaps of the four heuristics for turnover-based storage with 20/40, 20/60, and 20/80 demand skewness. **Figure 5.9** to Figure 5.11 compare the performance of the four heuristics with different demand skewness patterns. For the *LR* policy, the figures indicate that the percentage optimality gap decreases when the warehouse size increases. This result is in contrast to the case of random storage assignment, where an increase in the warehouse size led to an increase in the optimality gap for this heuristic. Furthermore, an increase in the skewness of demand reduces the optimality gap of the heuristic. The gap for each warehouse size decreases as the demand skewness increases. This result is due to the fact that in case of higher demand skewness, frequently requested items are assigned closer to the depot, resulting in shorter travel distances for the *LR* heuristic. The performance of the *LS* policy, in contrast, gets worse for high demand skewness (e.g. 20/60 or 20/80). Even though higher demand skewness moves frequently requested items closer to the depot, items requested on a particular pick list may still be scattered across different aisles. Since the picker must traverse each aisle containing a requested item completely if the *LS* heuristic is applied, s/he often needs to pass through every aisle resulting in very long travel distances. This situation becomes worse when the warehouse is bigger.

Summarizing the insights obtained from Figure 5.8 to Figure 5.11 combined, we conclude that the turnover-based storage assignment together with the *LR* routing policy is a very good combination for large warehouses; otherwise, the *LS* routing policy performs very well on every other case.

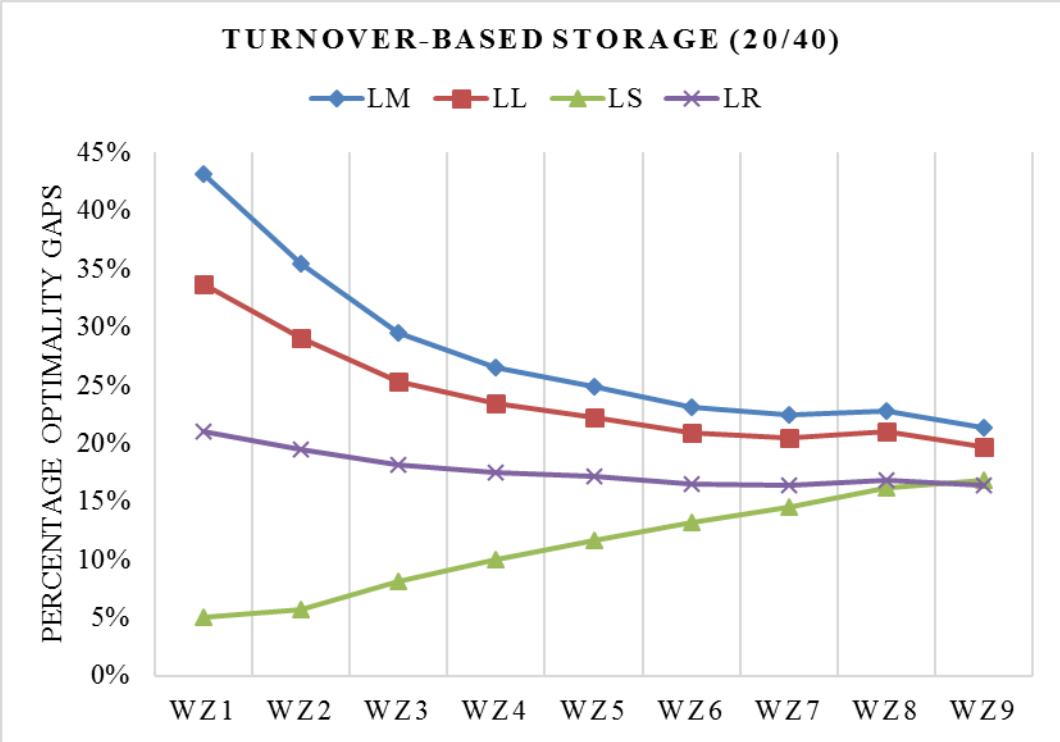


Figure 5.9 Average percentage optimality gaps of the heuristics under different warehouse sizes using turnover-based storage (20/40).

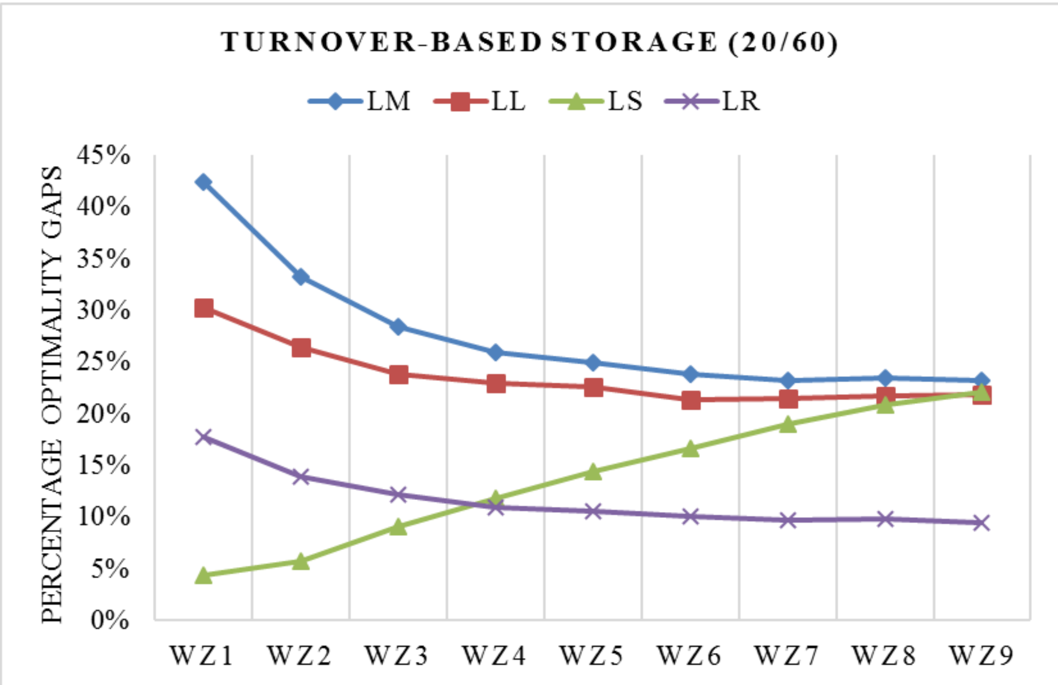


Figure 5.10 Average percentage optimality gaps of the heuristics under different warehouse sizes using turnover-based storage (20/60).

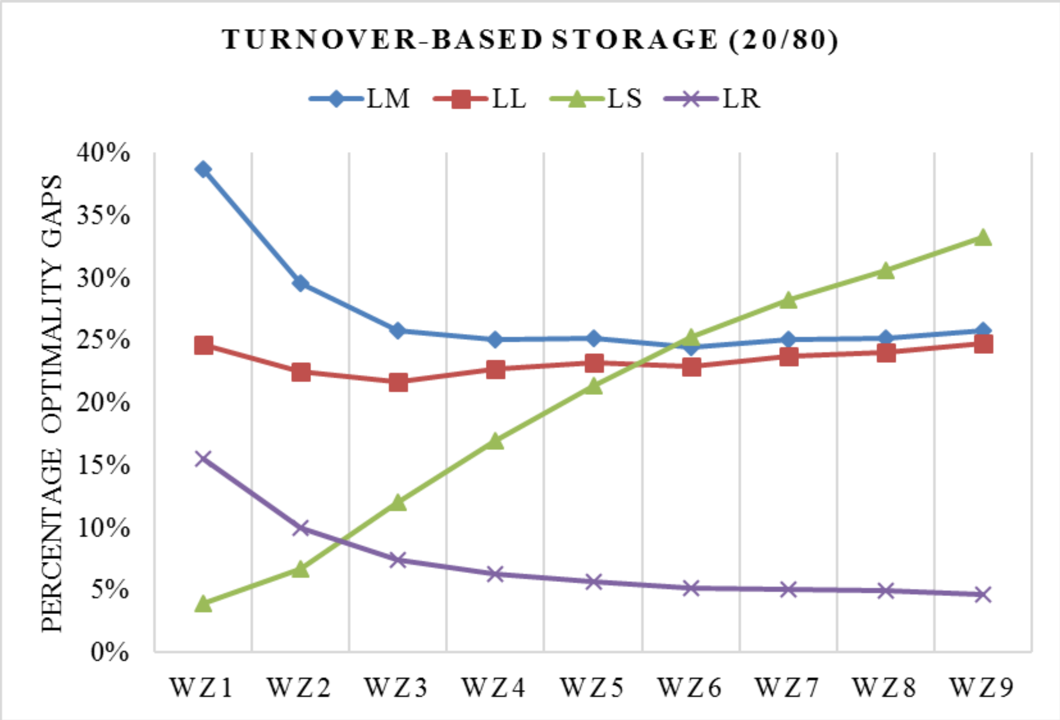


Figure 5.11 Average percentage optimality gaps of the heuristics under different warehouse sizes using turnover-based storage (20/80).

5.6 Conclusion

This chapter proposed an exact routing algorithm based on the algorithms developed by RR and Roodbergen and de Koster (2001a). Moreover, we proposed four simple routing heuristics, referred to as the *leaf S-shape*, *leaf return*, *leaf midpoint*, and *leaf largest gap* heuristics. We evaluated the performance of these heuristics compared to the exact algorithm for various storage assignment policies. Our computational results showed that the picking tours resulting from the routing heuristics were, on average, between 3.96% to 43.68% longer than the picking tours generated by the exact algorithm. These findings encourage practitioners to use exact routing algorithm in practice since it results in tour lengths that are significantly shorter than those generated by heuristics. The drawback of the exact algorithm is that it may generate a complex route, which might confuse the order picker if no proper guiding system is used. In that case, simple routing heuristics can be applied, and our results imply that turnover-based storage assignment together with the *leaf return* heuristic lead to good results for large leaf warehouses, while the *leaf S-shape* heuristic performs very well on other cases. When deciding on which picker routing policies to apply in practice, practitioners must evaluate the advantages and disadvantages of the exact and routing heuristics in their warehouse.

This work could be extended into various directions. For example, future research could study leaf warehouses with multiple depots or other non-conventional warehouses that have not attracted much attention in picker routing yet, such as the butterfly warehouse (Öztürkoğlu et al., 2012). Future work could also study the effect of order batching on the performance of the proposed routing policies or investigate picker routing in the leaf warehouse in a situation where pick lists can be updated during the pick process. We leave these and other extensions for future research.

Chapter 6 Conclusion

Order picking is one of the most critical operations for every supply chain because of its direct influence on customer satisfaction and the high time investment that is usually required for completing it. Hence, improving order picking efficiency can enhance both customer satisfaction and warehouse throughput. One of the major activities in order picking is the travelling of order pickers through the warehouse for retrieving items contained on pick-lists, which accounts for a large share of the total warehouse operating cost. To support minimizing this cost, this dissertation developed several efficient order picker routing policies for manual picker-to-parts order picking systems. Chapter 2 of this dissertation first conducted a systematic review of research on order picker routing. We proposed a conceptual framework of order picker routing and characterized the existing literature following this framework. Routing algorithms were categorized according to their type (exact, heuristic, and meta-heuristic) and the investigated warehouse layout (conventional, non-conventional, and general). One result of the literature review is that most earlier works assumed that picking tours start and end at the same location, which is usually the depot. However, order picking tours in practice do not necessarily start and end at the depot, for example in case tours are updated in real time while they are being completed, which is common in an e-commerce environment, for example. Chapter 3 therefore developed an exact and a heuristic routing procedure with arbitrary starting and ending points of a picking tour for the two-block warehouse. The exact algorithm proposed in this chapter is based on the concepts of Ratliff and Rosenthal (1983) as well as Roodbergen and de Koster (2001a) that used graph theory and a dynamic programming procedure. We also developed a routing heuristic, denoted *S*-shape*, and evaluated its performance compared to the exact algorithm. Based on our experiments, the exact algorithm obtained tours that were between 6.32% and 35.34% shorter than those generated by *S*-shape* heuristic.

Another result obtained in Chapter 2 is that the order picker routing problem in non-conventional warehouses has not received much attention so far. Motivated by this observation, Chapter 4 proposed an exact algorithm for the routing of order pickers through the chevron warehouse. We investigated the effect of different storage assignment policies on the order picking tour obtained by the proposed exact algorithm, and further modified two simple routing heuristics, referred to as the *chevron midpoint* and *chevron largest gap* heuristics. The average order picking tour lengths resulting from the exact algorithm and the heuristics were compared to evaluate the performance of the routing heuristics under various storage assignment policies used in warehouses. The results indicate that the tours resulting from the exact algorithm are 10.29% to 39.08% shorter than the tours generated by the routing heuristics. Chapter 5 proposed an exact algorithm as well as four simple routing heuristics, referred to as the *leaf S-shape*, *leaf return*, *leaf midpoint*, and *leaf largest gap* heuristics, for solving the routing problem in the leaf warehouse. We evaluated the performance of these heuristics compared to the exact algorithm for various storage assignment policies. Our computational results showed that the picking tours resulting from the heuristics were, on average, 3.96% to 43.68% longer than the picking tours generated by the exact algorithm. The exact algorithms proposed in Chapters 4 and 5 were again based on the concepts of Ratliff and Rosenthal (1983) as well as and Roodbergen and de Koster (2001a).

The results obtained in this dissertation emphasize that optimal order picker routing policies should be the preferred means of guiding the order picker through the warehouse. Even though using exact algorithm may lead to difficulties in practical applications as order pickers may get confused by the optimal routes, we can see nowadays that more and more modern communication equipment is being used in warehouses, such as tablets, pick-by-voice or pick-by-light applications. These devices make it possible to communicate optimal routes in a simple manner to the order picker, such that it is not necessary anymore (for example) to partially memorize a complex tour. These modern devices can easily be connected to warehouse management software. We therefore believe that modern communication technology supports the use of our proposed exact algorithms in practice.

The dissertation at hand has some limitations. In Chapter 2, we considered only papers for the literature sample and the subsequent analysis that were published in peer-reviewed journals, and we excluded papers that appeared in other outlets (e.g. book chapters or conference proceedings). These filters may have led to the exclusion of relevant works. Therefore, future literature reviews could analyze papers published in these outlets as well. Moreover, Chapter 2 reviewed order picker routing for manual picker-to-parts systems without considering the routing of robots in automated warehouses. Future research could further investigate the routing of robots to gain additional insights into the routing problem in warehousing. In terms of Chapters 3 to 5, there are several options for extending the current works. First, we assumed for all three scenarios we investigated that order picker congestion cannot occur within aisles. Future research could consider multiple order pickers picking at the same time, which could lead to picker congestion, and investigate how congestion influences optimal picker routing. Secondly, future research could investigate the case of a dynamic order picking system where a pick-list can be updated any time. While the algorithm proposed in Chapter 3 is generally compatible with such a scenario, the methods proposed in Chapters 4 and 5 would have to be adjusted. Furthermore, developing exact algorithms and routing heuristics for the routing of order pickers in other non-conventional warehouses, such as the butterfly warehouse, would be an interesting topic for the future research.

Appendix

List of papers included in the extended sample in Chapter 2 that were not cited in the text.

Albareda-Sambola, M., Alonso-Ayuso, A., Molina, E., & De Blas, C. S. (2009). Variable neighborhood search for order batching in a warehouse. *Asia-Pacific Journal of Operational Research*, *26*(05), 655-683.

Azadnia, A. H., Taheri, S., Ghadimi, P., Mat Saman, M. Z., & Wong, K. Y. (2013). Order batching in warehouses by minimizing total tardiness: a hybrid approach of weighted association rule mining and genetic algorithms. *The Scientific World Journal*, *2013*. 1-13.

Battini, D., Glock, C. H., Grosse, E. H., Persona, A., & Sgarbossa, F. (2016). Human energy expenditure in order picking storage assignment: A bi-objective method. *Computers & Industrial Engineering*, *94*, 147-157.

Berglund, P., & Batta, R. (2012). Optimal placement of warehouse cross-aisles in a picker-to-part warehouse with class-based storage. *IIE Transactions*, *44*(2), 107-120.

Bindi, F., Manzini, R., Pareschi, A., & Regattieri, A. (2009). Similarity-based storage allocation rules in an order picking system: an application to the food service industry. *International Journal of Logistics: Research and Applications*, *12*(4), 233-247.

Bottani, E., Cecconi, M., Vignali, G., & Montanari, R. (2012). Optimisation of storage allocation in order picking operations through a genetic algorithm. *International Journal of Logistics Research and Applications*, *15*(2), 127-146.

Bozer, Y. A., & Kile, J. W. (2008). Order batching in walk-and-pick order picking systems. *International Journal of Production Research*, *46*(7), 1887-1909.

Caron, F., Marchet, G., & Perego, A. (1998). Routing policies and COI-based storage policies in picker-to-part systems. *International Journal of Production Research*, *36*(3), 713-732.

Caron, F., Marchet, G., & Perego, A. (2000). Layout design in manual picking systems: a simulation approach. *Integrated Manufacturing Systems*, *11*(2), 94-104.

Chackelson, C., Errasti, A., Ciprés, D., & Lahoz, F. (2013). Evaluating order picking performance trade-offs by configuring main operating strategies in a retail distributor: A Design of Experiments approach. *International Journal of Production Research*, *51*(20), 6097-6109.

Chan, F. T., & Chan, H. K. (2011). Improving the productivity of order picking of a manual-pick and multi-level rack distribution warehouse through the implementation of class-based storage. *Expert Systems with Applications*, *38*(3), 2686-2700.

Chen, C. M., Gong, Y., De Koster, R. B., & Van Nunen, J. A. (2010). A flexible evaluative framework for order picking systems. *Production and Operations Management*, *19*(1), 70-82.

Chen, M. C., & Wu, H. P. (2005). An association-based clustering approach to order batching considering customer demand patterns. *Omega*, *33*(4), 333-343.

Chiang, D. M. H., Lin, C. P., & Chen, M. C. (2011). The adaptive approach for storage assignment by mining data of warehouse management system for distribution centres. *Enterprise Information Systems*, *5*(2), 219-234.

Chuang, Y. F., Chia, S. H., & Wong, J. Y. (2014). Enhancing order-picking efficiency through data mining and assignment approaches. *WSEAS Transactions on Business and Economics*, *11*(1), 52-64.

Dekker, R., De Koster, M. B. M., Roodbergen, K. J., & Van Kalleveen, H. (2004). Improving order-picking response time at Ankor's warehouse. *Interfaces*, *34*(4), 303-313.

Diaz, R. (2016). Using dynamic demand information and zoning for the storage of non-uniform density stock keeping units. *International Journal of Production Research*, *54*(8), 2487-2498.

Dijkstra, A. S., & Roodbergen, K. J. (2017). Exact route-length formulas and a storage location assignment heuristic for picker-to-parts warehouses. *Transportation Research Part E: Logistics and Transportation Review*, *102*, 38-59.

Đukić, G., & Oluić, Č. (2004). Heuristics, Advanced Heuristics or Optimal Algorithm. *Strojniški vestnik*, *50*(11), 530-535.

Đukić, G., & Oluić, Č. (2007). Order-picking methods: improving order-picking efficiency. *International Journal of Logistics Systems and Management*, *3*(4), 451-460.

Giannikas, V., Lu, W., Robertson, B., & McFarlane, D. (2017). An interventionist strategy for warehouse order picking: Evidence from two case studies. *International Journal of Production Economics*, *189*, 63-76.

Gómez-Montoya, R. A., Correa-Espinal, A. A., & Hernández-Vahos, J. D. (2016). Picking Routing Problem with K homogenous material handling equipment for a refrigerated warehouse. *Revista Facultad de Ingeniería Universidad de Antioquia*, *80*, 9-20.

Guo, J., Zhou, L., & Zhu, J. (2011). Stochastic model and simulation research for random-storage S-type manual order picking. *Key Engineering Materials*, *467*, 92-97.

Guo, L., & Yang, X. J. (2014). Order Batching Picking Based on Activity Based Classification Storage. *Advanced Materials Research*, *981*, 957-960.

Henn, S. (2015). Order batching and sequencing for the minimization of the total tardiness in picker-to-part warehouses. *Flexible Services and Manufacturing Journal*, *27*(1), 86-114.

Henn, S., & Wäscher, G. (2012). Tabu search heuristics for the order batching problem in manual order picking systems. *European Journal of Operational Research*, *222*(3), 484-494.

Ho, Y. C., & Liu, C. F. (2005). A design methodology for converting a regular warehouse into a zone-picking warehouse. *Journal of the Chinese Institute of Industrial Engineers*, *22*(4), 332-345.

Ho, Y. C., Su, T. S., & Shi, Z. B. (2008). Order-batching methods for an order-picking warehouse with two cross aisles. *Computers & Industrial Engineering*, *55*(2), 321-347.

Hong, S., & Kim, Y. (2017). A route-selecting order batching model with the S-shape routes in a parallel-aisle order picking system. *European Journal of Operational Research*, *257*(1), 185-196.

Hong, S., Johnson, A. L., & Peters, B. A. (2012). Large-scale order batching in parallel-aisle picking systems. *IIE Transactions*, *44*(2), 88-106.

Hsieh, L. F., & Huang, Y. C. (2011). New batch construction heuristics to optimise the performance of order picking systems. *International Journal of Production Economics*, *131*(2), 618-630.

Hwang, H., & Kim, D. G. (2005). Order-batching heuristics based on cluster analysis in a low-level picker-to-part warehousing system. *International Journal of Production Research*, *43*(17), 3657-3670.

Hwang, H., Oh, Y. H., & Lee, Y. K. (2004). An evaluation of routing policies for order-picking operations in low-level picker-to-part system. *International Journal of Production Research*, *42*(18), 3873-3889.

Jakubiak, M., & Tarczyński, G. (2012). Selection of manual order picking concepts in a warehouse by means of simulation tools. *Mathematical Economics*, *8* (15), 45-62.

Kłodawski, M., & Żak, J. (2013). Order picking area layout and its impact on efficiency of order picking process. *Journal of Traffic and Logistics Engineering*, *1*(1), 41-46.

Koch, S., & Wäscher, G. (2016). A grouping genetic algorithm for the Order Batching Problem in distribution warehouses. *Journal of Business Economics*, *86*(1-2), 131-153.

Kuo, R. J., Shieh, M. C., Zhang, J. W., & Chen, K. Y. (2013). The application of an artificial immune system-based back-propagation neural network with feature selection to an RFID positioning system. *Robotics and Computer-Integrated Manufacturing*, *29*(6), 431-438.

Le-Duc, T., & De Koster, R. M. B. (2005). Travel distance estimation and storage zone optimization in a 2-block class-based storage strategy warehouse. *International Journal of Production Research*, *43*(17), 3561-3581.

Mantel, R. J., Schuur, P. C., & Heragu, S. S. (2007). Order oriented slotting: a new assignment strategy for warehouses. *European Journal of Industrial Engineering*, *1*(3), 301-316.

Manzini, R., Gamberi, M., Persona, A., & Regattieri, A. (2007). Design of a class based storage picker to product order picking system. *The International Journal of Advanced Manufacturing Technology*, *32*(7-8), 811-821.

Melacini, M., Perotti, S., & Tumino, A. (2011). Development of a framework for pick-and-pass order picking system design. *The International Journal of Advanced Manufacturing Technology*, *53*(9-12), 841-854.

Menéndez, B., Bustillo, M., Pardo, E. G., & Duarte, A. (2017). General Variable Neighborhood Search for the Order Batching and Sequencing Problem. *European Journal of Operational Research*, *263*(1), 82-93.

Menéndez, B., Pardo, E. G., Duarte, A., Alonso-Ayuso, A., & Molina, E. (2015). General Variable Neighborhood Search applied to the picking process in a warehouse. *Electronic Notes in Discrete Mathematics*, *47*, 77-84.

Menéndez, B., Pardo, E. G., Sánchez-Oro, J., & Duarte, A. (2017). Parallel variable neighborhood search for the min–max order batching problem. *International Transactions in Operational Research*, *24*(3), 635-662.

Ming-Huang Chiang, D., Lin, C. P., & Chen, M. C. (2014). Data mining based storage assignment heuristics for travel distance reduction. *Expert Systems*, *31*(1), 81-90.

Muter, İ., & Öncan, T. (2015). An exact solution approach for the order batching problem. *IIE Transactions*, *47*(7), 728-738.

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

Pan, J. C. H., & Shih, P. H. (2008). Evaluation of the throughput of a multiple-picker order picking system with congestion consideration. *Computers & Industrial Engineering*, *55*(2), 379-389.

Pan, J. C. H., & Wu, M. H. (2012). Throughput analysis for order picking system with multiple pickers and aisle congestion considerations. *Computers & Operations Research*, *39*(7), 1661-1672.

Pan, J. C. H., Wu, M. H., & Chang, W. L. (2014). A travel time estimation model for a high-level picker-to-part system with class-based storage policies. *European Journal of Operational Research*, *237*(3), 1054-1066.

Parikh, P. J., & Meller, R. D. (2009). Estimating picker blocking in wide-aisle order picking systems. *IIE Transactions*, *41*(3), 232-246.

Parikh, P. J., & Meller, R. D. (2010). A note on worker blocking in narrow-aisle order picking systems when pick time is non-deterministic. *IIE Transactions*, *42*(6), 392-404.

Pérez-Rodríguez, R., Hernández-Aguirre, A., & Jöns, S. (2015). A continuous estimation of distribution algorithm for the online order-batching problem. *The International Journal of Advanced Manufacturing Technology*, *79*(1-4), 569-588.

Petersen, C. G. (2002). Considerations in order picking zone configuration. *International Journal of Operations & Production Management*, *22*(7), 793-805.

Petersen, C. G., Siu, C., & Heiser, D. R. (2005). Improving order picking performance utilizing slotting and golden zone storage. *International Journal of Operations & Production Management*, *25*(10), 997-1012.

Qin, K., Chen, F. Y., & Ma, L. (2015). Cutting down the travel distance of put systems at Kunming International Flower Auction Market. *International Journal of Production Research*, *53*(12), 3573-3585.

Rakesh, V., & Adil, G. K. (2015). Layout optimization of a three dimensional order picking warehouse. *IFAC-PapersOnLine, 48*(3), 1155-1160.

Rao, S. S., & Adil, G. K. (2013). Class-based storage with exact S-shaped traversal routeing in low-level picker-to-part systems. *International Journal of Production Research*, *51*(16), 4979-4996.

Rao, S. S., & Adil, G. K. (2013). Optimal class boundaries, number of aisles, and pick list size for low-level order picking systems. *IIE Transactions*, *45*(12), 1309-1321.

Rao, S. S., & Adil, G. K. (2017). Analytical models for a new turnover-based hybrid storage policy in unit-load warehouses. *International Journal of Production Research*, *55*(2), 327-346.

Rubrico, J. I. U., Ota, J., Higashi, T., & Tamura, H. (2008). Metaheuristic scheduling of multiple picking agents for warehouse management. *Industrial Robot: An International Journal*, *35*(1), 58-68.

Sadowsky, V., & Ten Hompel, M. (2011). Calculation of the average travel distance in a low-level picker-to-part system considering any distribution function within the aisles. *Logistics Journal*, *2011*(01), 1-28.

Shqair, M., Altarazi, S., & Al-Shihabi, S. (2014). A statistical study employing agent-based modeling to estimate the effects of different warehouse parameters on the distance traveled in warehouses. *Simulation Modelling Practice and Theory*, *49*, 122-135.

Sooksaksun, N., Kachitvichyanukul, V., & Gong, D. C. (2012). A class-based storage warehouse design using a particle swarm optimisation algorithm. *International Journal of Operational Research*, *13*(2), 219-237.

Tarczyński, G. (2012). Analysis of the impact of storage parameters and the size of orders on the choice of the method for routing order picking. *Operations Research and Decisions*, *22*(4), 105-120.

Tarczyński, G. (2015). Estimating order-picking times for return heuristic-equations and simulations. *LogForum*, *11*(3), 295-303.

Tarczyński, G. (2017). The impact of COI-based storage on order-picking times. *LogForum*, *13*(3), 313-326.

Thomas, L. M., & Meller, R. D. (2014). Analytical models for warehouse configuration. *IIE Transactions*, *46*(9), 928-947.

Valle, C. A., Beasley, J. E., & da Cunha, A. S. (2017). Optimally solving the joint order batching and picker routing problem. *European Journal of Operational Research*, *262*(3), 817-834.

Van Den Berg, J. P., & Van Der Hoff, H. H. (2001). An order batching algorithm for wave picking in a parallel-aisle warehouse. *IIE Transactions*, *33*(5), 385-398.

Van Gils, T., Ramaekers, K., Braekers, K., Depaire, B., & Caris, A. (2018). Increasing order picking efficiency by integrating storage, batching, zone picking, and routing policy decisions. *International Journal of Production Economics*, *197*, 243-261.

Venkitasubramony, R., & Adil, G. K. (2017). Design of an order-picking warehouse factoring vertical travel and space sharing. *The International Journal of Advanced Manufacturing Technology*, *91*(5-8), 1921-1934.

Vis, I. F., & Roodbergen, K. J. (2011). Layout and control policies for cross docking operations. *Computers & Industrial Engineering*, *61*(4), 911-919.

Wruck, S., Vis, I. F., & Boter, J. (2013). Time-restricted batching models and solution approaches for integrated forward and return product flow handling in warehouses. *Journal of the Operational Research Society*, *64*(10), 1505-1516.

Xiao, J., & Zheng, L. (2010). A correlated storage location assignment problem in a single-block-multi-aisles warehouse considering BOM information. *International Journal of Production Research*, *48*(5), 1321-1338.

Zhang, J., Wang, X., & Huang, K. (2016). Integrated on-line scheduling of order batching and delivery under B2C e-commerce. *Computers & Industrial Engineering*, *94*, 280-289.

Zhang, J., Wang, X., & Huang, K. (2018). On-line scheduling of order picking and delivery with multiple zones and limited vehicle capacity. *Omega*, *79*, 104-115.

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

Zhu, J., Guo, J., & Zhou, L. (2011). Research on stochastic model and simulation for sorted-storage S-type manual order picking. *Key Engineering Materials*, *467*, 98-103.

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

Theorem A.1 (Ratliff and Rosenthal, 1983) In a conventional warehouse with a single block, a subgraph $T_j \subset L_j$ (either L_j^- or L_j^+) is an L_j PTS if and only if all of the following conditions hold.

1. The degrees of all vertices v_i of L_j, representing item locations, are positive in T_j.
2. Every vertex in T_j, except possibly for a_j and b_j, has even or zero degree.
3. Excluding vertices of zero degree, T_j has either
    - no connected component,
    - a single connected component containing at least one of a_j and b_j,
    - two connected components, with each component containing at least one of a_j and b_j.

Theorem A.2 (Roodbergen and de Koster, 2001a) In a conventional warehouse with two blocks, a subgraph $T_j \subset L_j$ (either L_j^-, L_j^{+l}, or L_j^{+u}) is an L_j PTS if and only if all of the following conditions hold.

1. The degrees of all vertices v_i of L_j, representing item locations, are positive in T_j.
2. Every vertex in T_j, except possibly for a_j, b_j, and c_j, has even or zero degree.
3. Excluding vertices of zero degree, T_j has either
    - no connected component,
    - a single connected component containing at least one of a_j, b_j, and c_j,
    - two connected components, with each component containing at least one of a_j, b_j, and c_j,
    - three connected components, with each component containing at least one of a_j, b_j, and c_j.

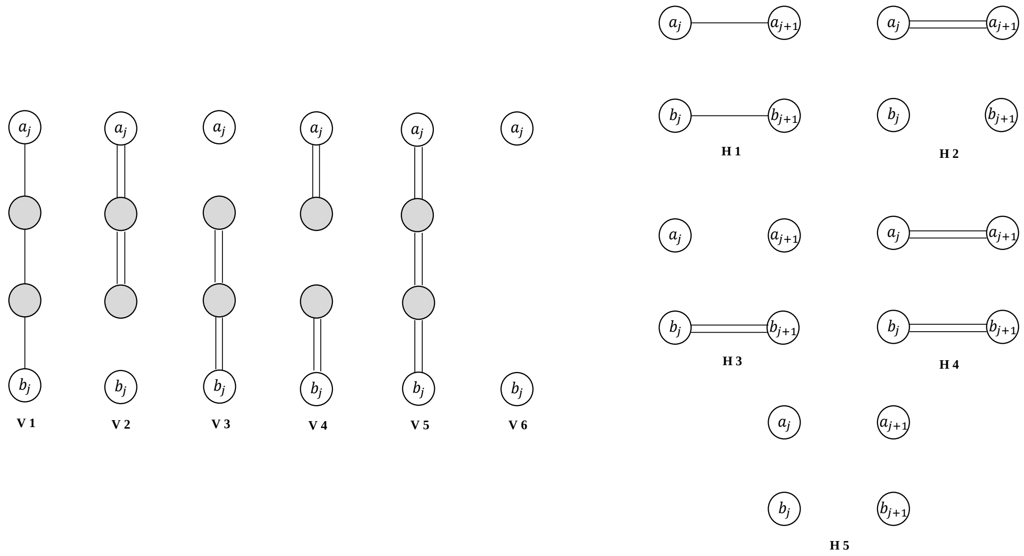


Figure A.1 Vertical and horizontal components for aisle j and travelling from aisle j to aisle $j+1$, respectively.

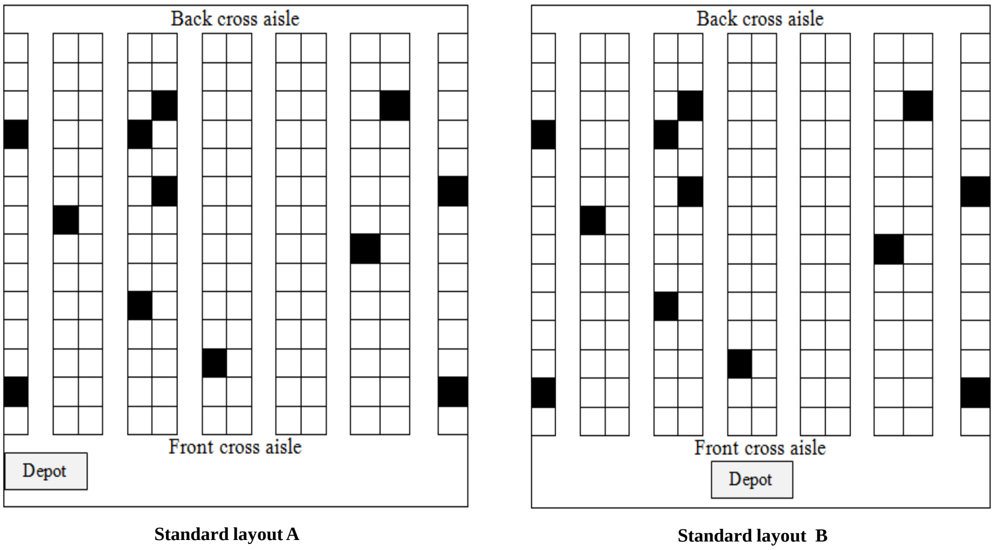


Figure A.2 Standard warehouse layouts.

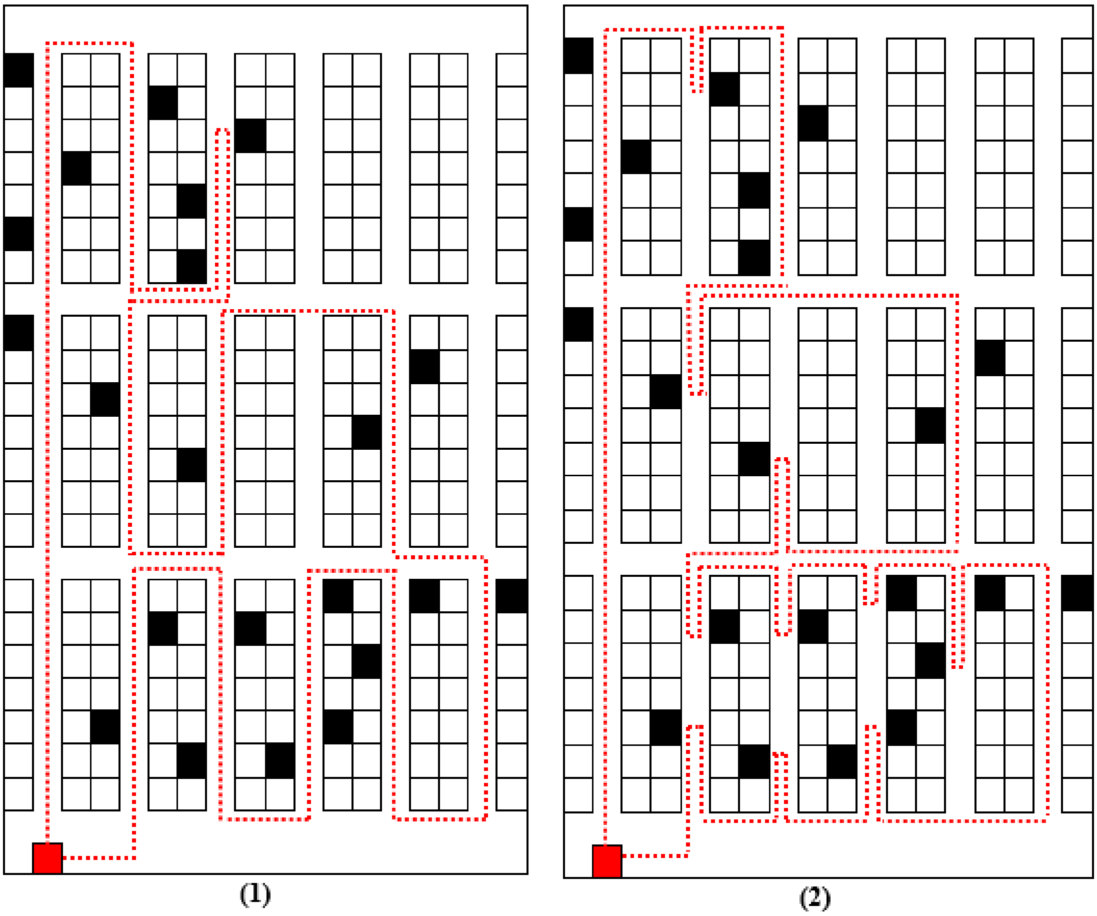


Figure A.3 *Multi-block S-shape* (1) and *Multi-block largest gap* (2) routing heuristics (Roodbergen and de Koster, 2001b)

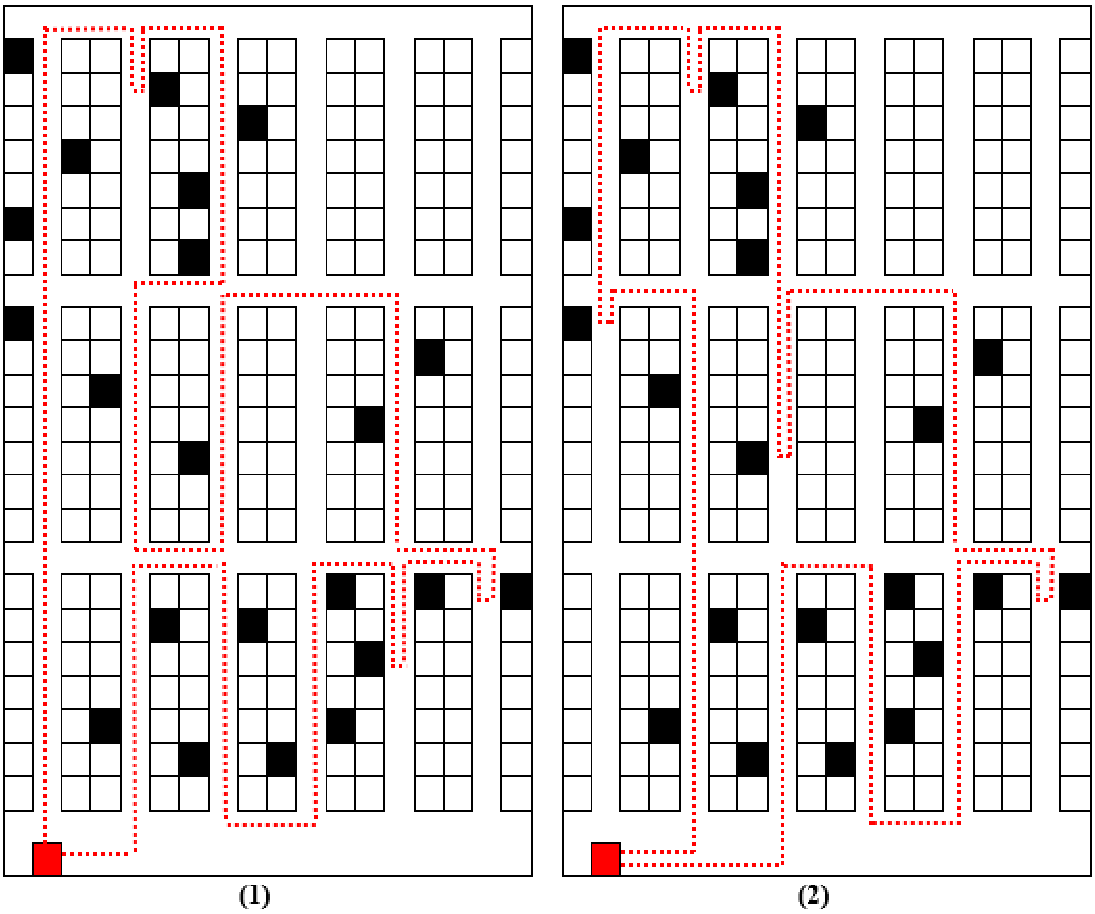


Figure A.4 *Combined* (1) and *Combined+* (2) routing heuristics (Roodbergen and de Koster, 2001b)

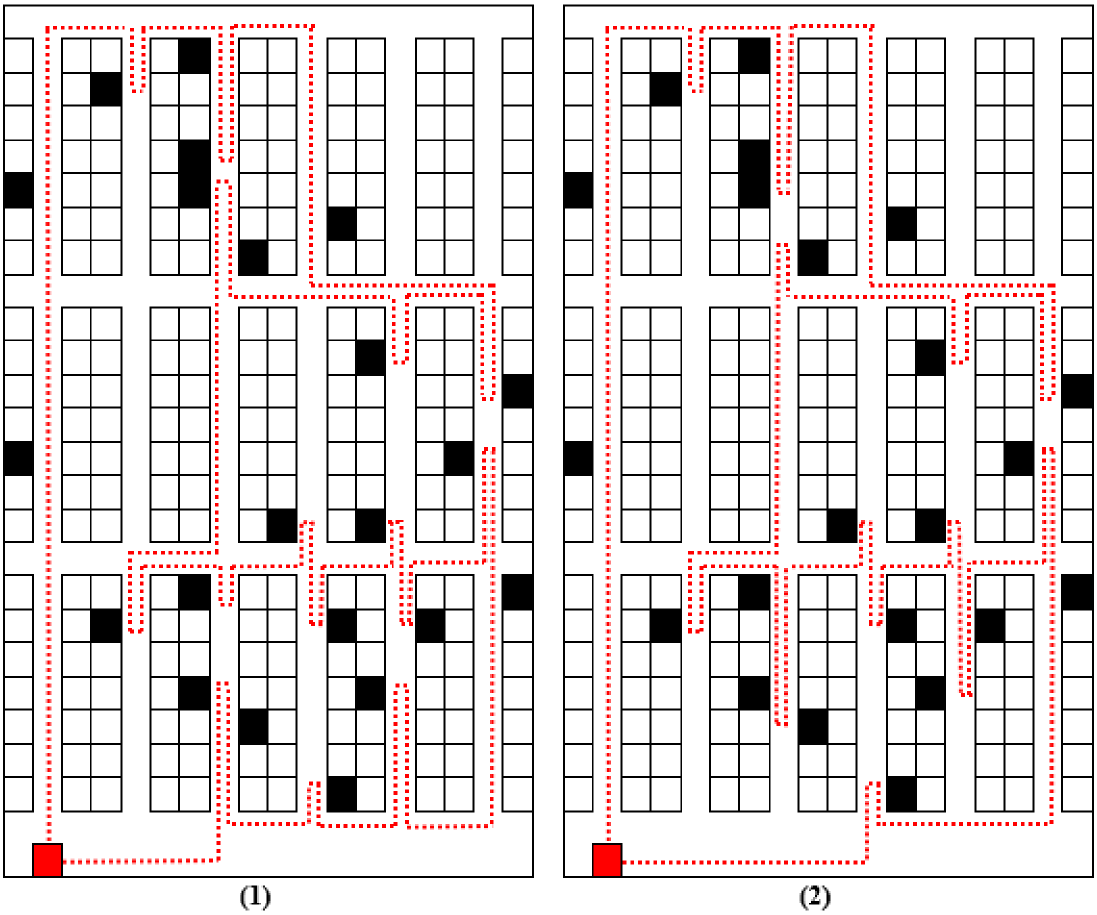


Figure A.5 *Block-aisle 1* (1) and *Block-aisle 2* (2) routing heuristics (Shouman et al., 2007)

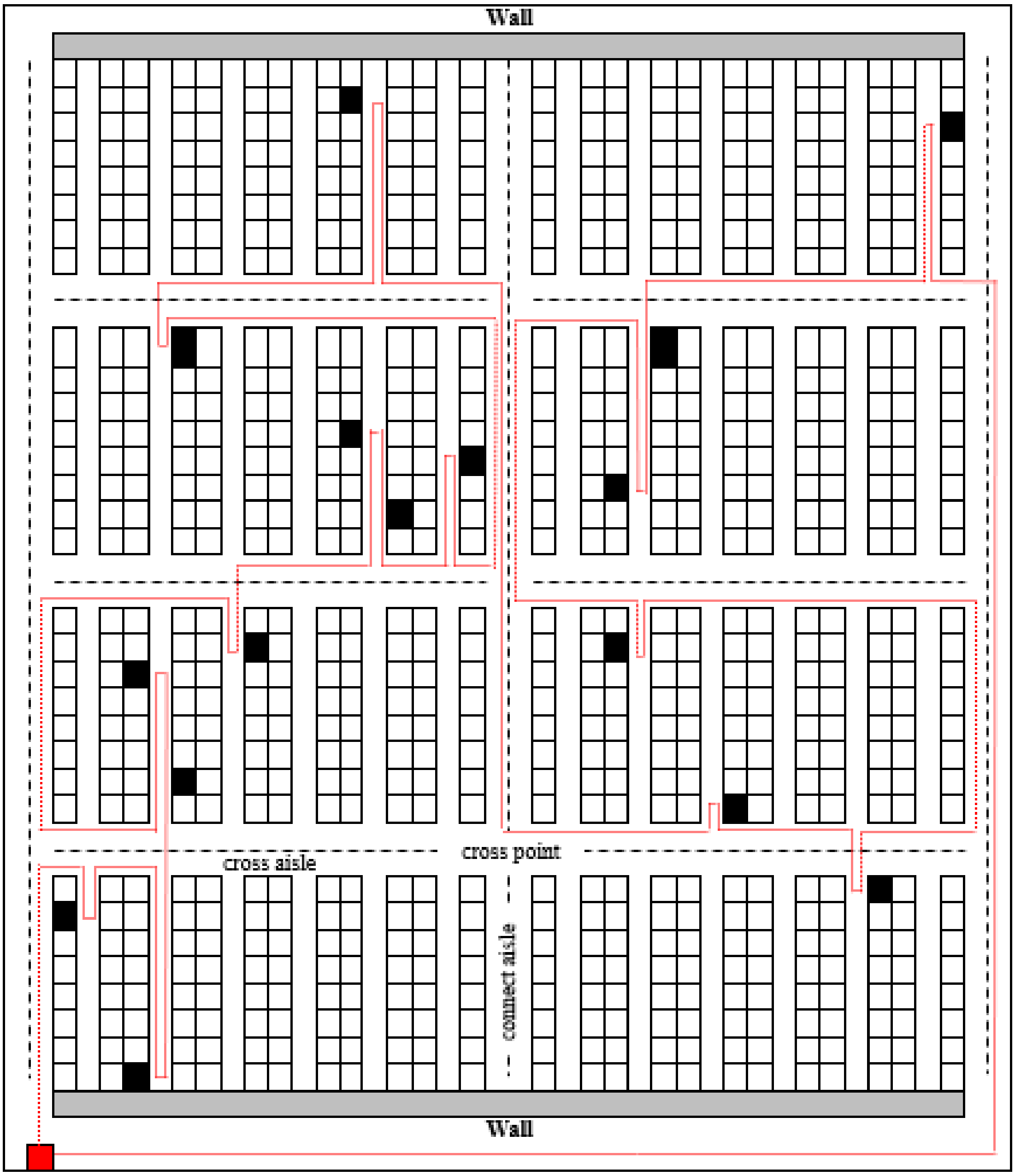


Figure A.6 *RNA* routing heuristic (Chen et al., 2019a)

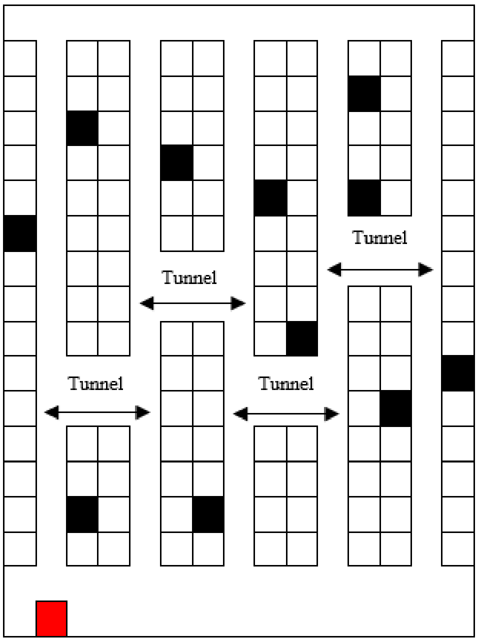


Figure A.7 A discrete cross aisle layout (Öztürkoğlu and Hoser, 2019)

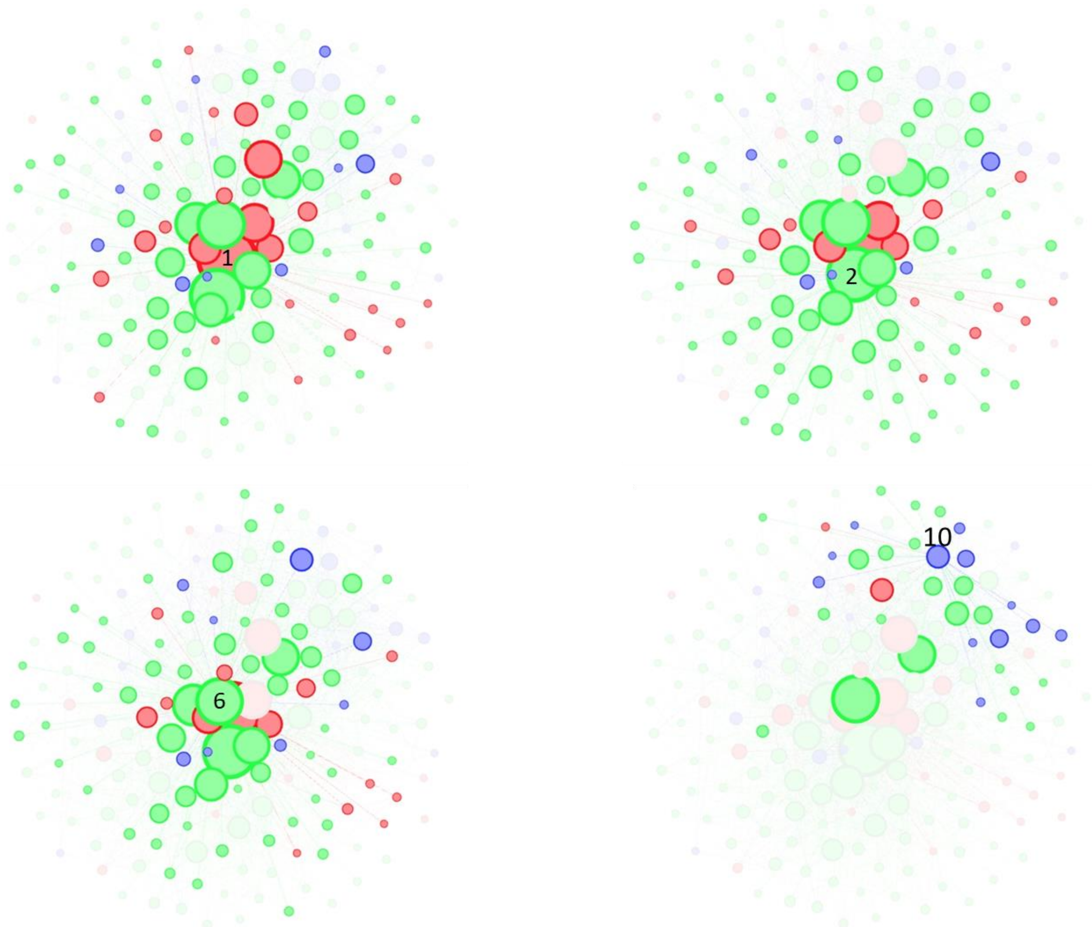


Figure A.8 Citation networks of 1) Ratliff and Rosenthal (1983), 2) Hall (1993), 6) Roodbergen and de Koster (2001b), and 10) Tsai et al. (2008).

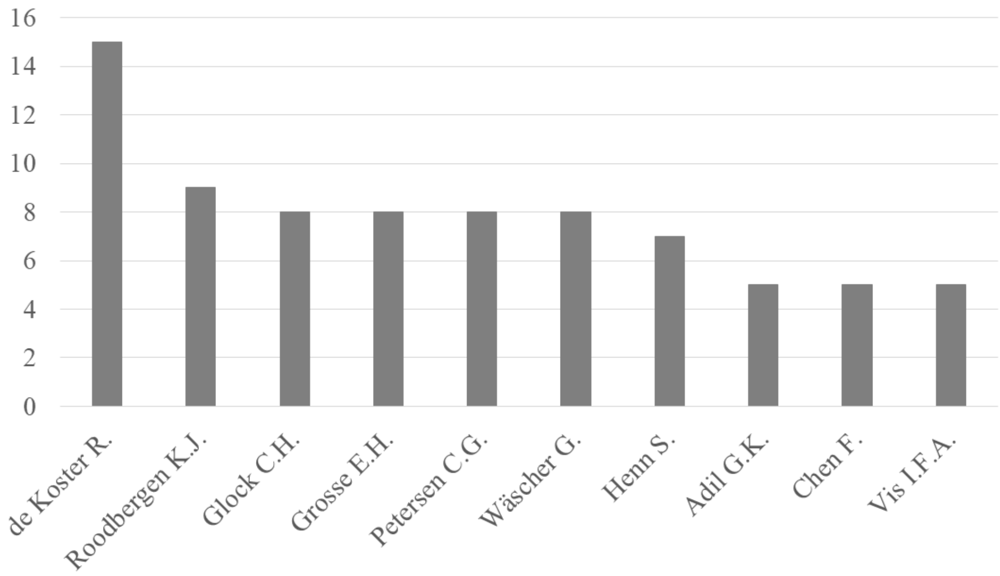


Figure A.9 Most contributing authors in the core and extended samples.

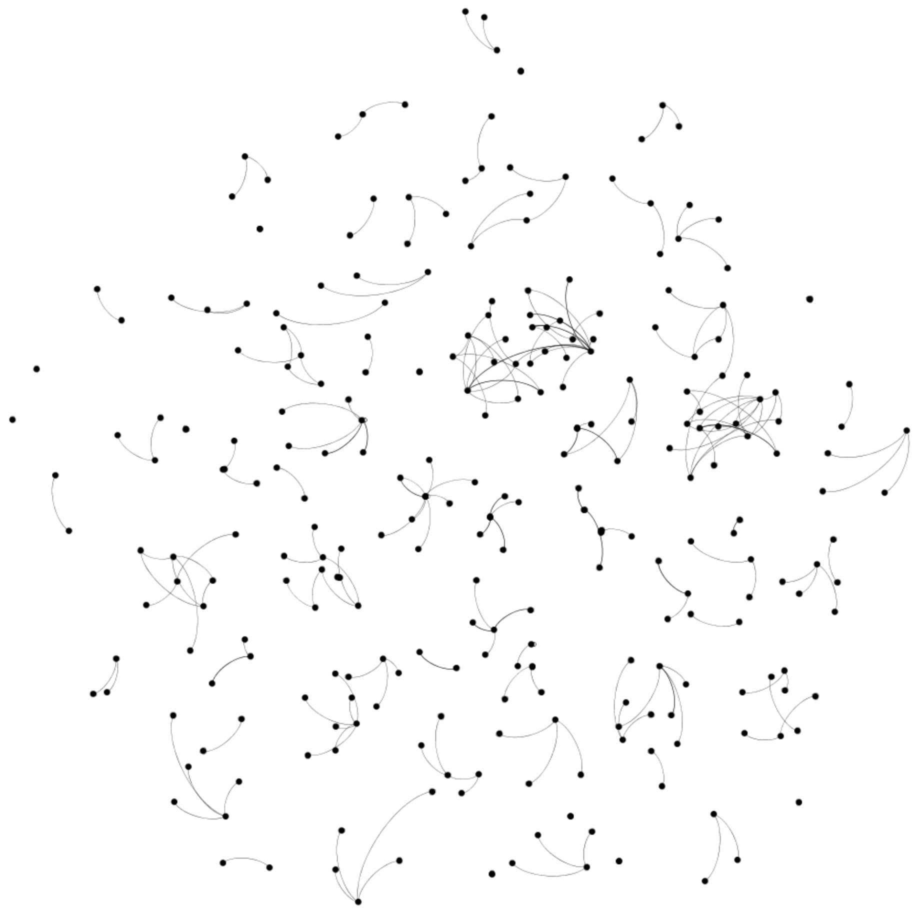


Figure A.10 Collaboration structure in the core and extended samples.

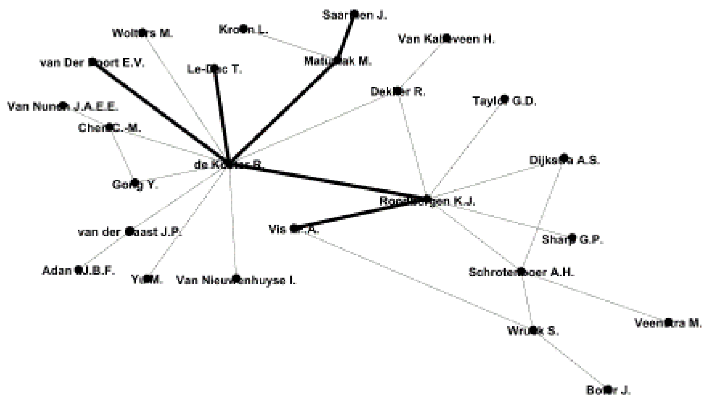


Figure A.11 Author cluster including de Koster and Roodbergen.

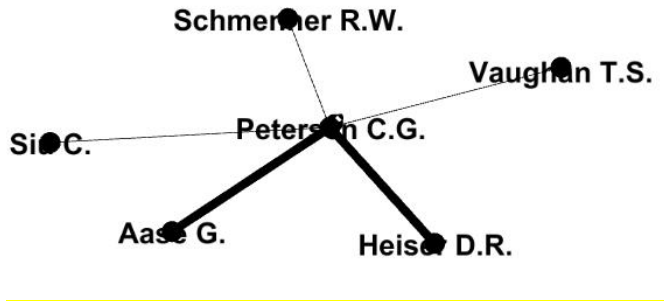


Figure A.12 Author cluster including Petersen.

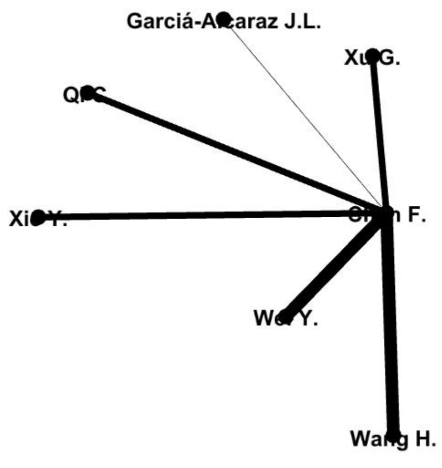


Figure A.13 Author cluster including Chen.

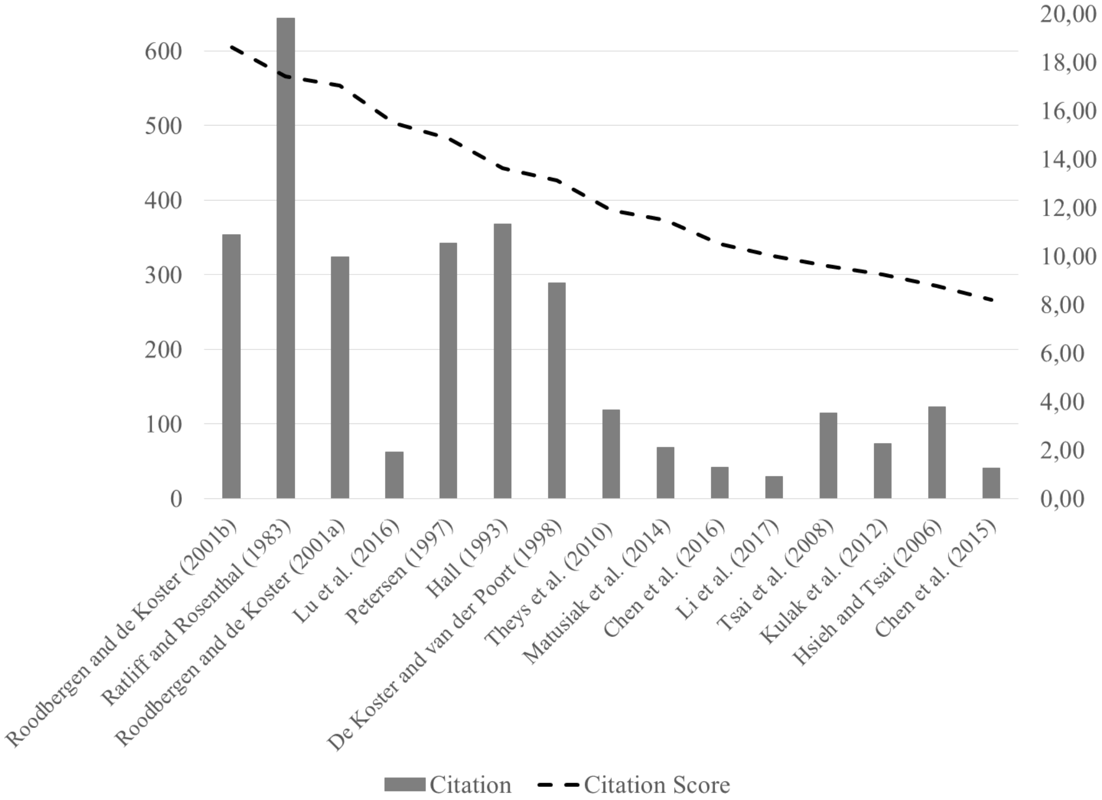


Figure A.14 Most frequently cited papers in the core sample[8].

[8] Figure A.14 shows both total number of citations (in Google Scholar by August 2019) and a citation score that takes into account the number of citations a paper received since publication divided by the number of years since publications. The papers a ranked according to the citation score.

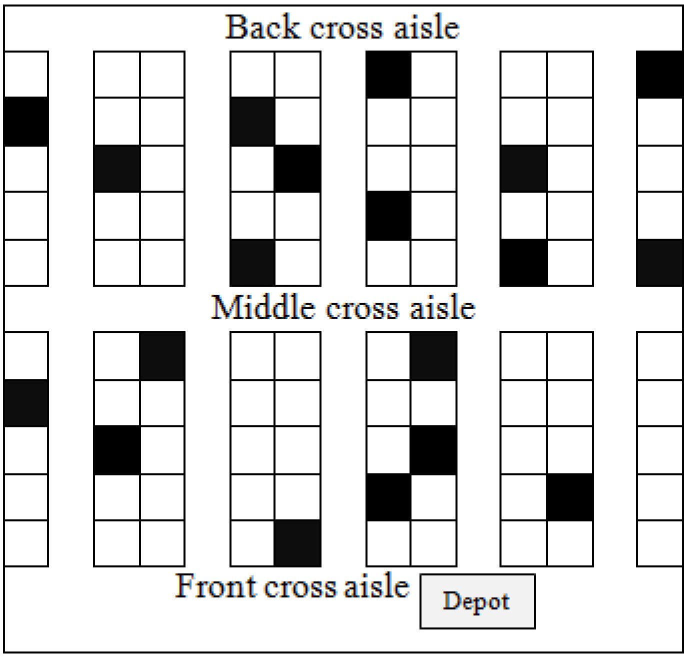


Figure A.15 Warehouse with two blocks.

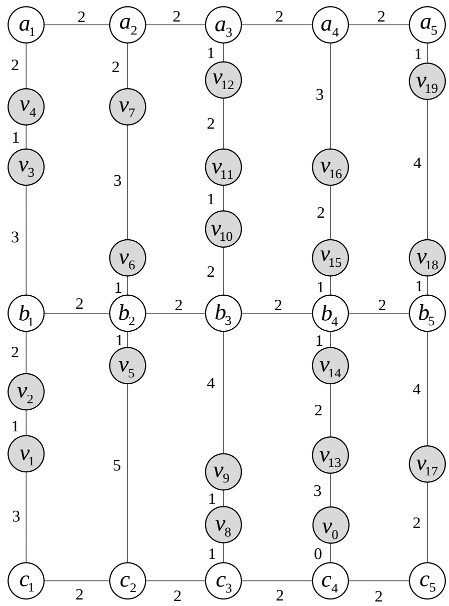


Figure A.16 Graph representation G of the warehouse in **Figure A.15**.

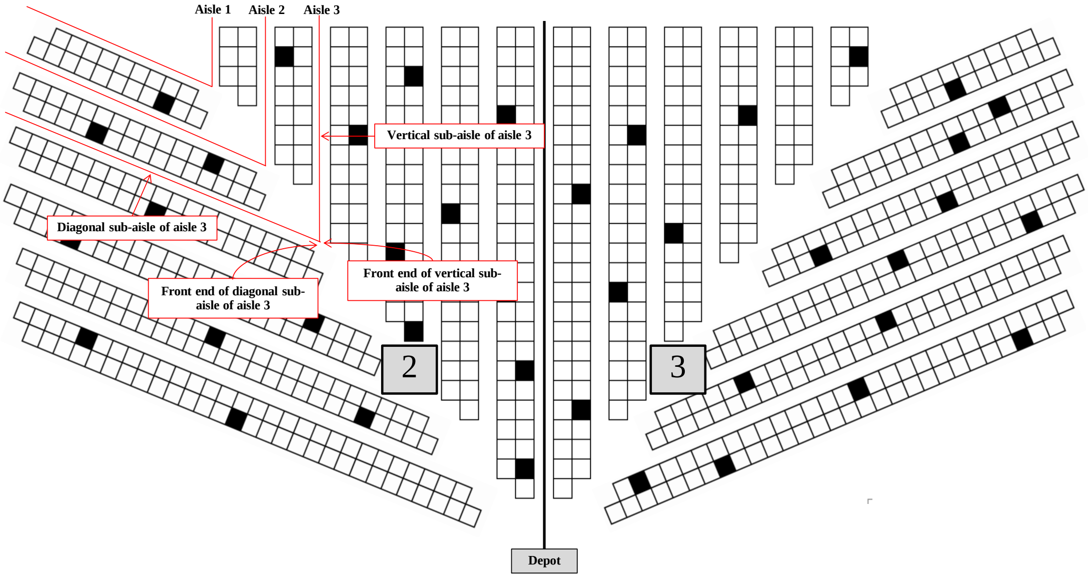


Figure A.17 Leaf warehouse.

Table A.1 Summary of the papers contained in the core sample.

Papers	Warehouse layout										
	Conventional		Non-conventional	General	Number of depots		Depot location(s) (*=not mentioned in the paper)	Narrow aisle	Wide aisle	Low-level storage racks	High-level storage racks
	Single-block	Multi-block			One	Multiple					
Ratliff and Rosenthal (1983)	x				x		depot in the middle of front cross aisle	x		x	
Goetschalckx and Ratliff (1988a)	x				x		depot in the middle of front cross aisle		x	x	
Goetschalckx and Ratliff (1988b)	x						*		x	x	
Hall (1993)	x				x		depot in the middle of front cross aisle	x		x	
Petersen (1997)	x				x		depot in the left corner of front cross aisle	x		x	
Singh and van Oudheusden (1997)				x	x		*	x		x	
Daniels et al. (1998)				x	x		*	x		x	
De Koster and van der Poort (1998)	x					x	multiple depot locations	x		x	
Vaughan and Petersen (1999)		x				x	multiple depot locations	x		x	
Roodbergen and de Koster (2001a)		x			x		depot in the middle of front cross aisle	x		x	
Roodbergen and de Koster (2001b)		x			x		depot in the left corner of front cross aisle	x		x	
Makris and Giakoumakis (2003)	x					x	multiple depot locations	x		x	
Ho and Tseng (2006)	x				x		depot in the left corner of front cross aisle	x		x	
Hsieh and Tsai (2006)		x				x	multiple depot locations	x		x	
Shouman et al. (2007)		x			x		depot in the left corner of front cross aisle	x		x	
Tsai et al. (2008)	x				x		depot in the left corner of front cross aisle	x		x	
Theys et al. (2010)		x			x		considered different depot locations	x		x	
Glock and Grosse (2012)			x		x		considered different depot locations		x	x	
Jang and Sun (2012)		x			x		*	x		x	
Kulak et al. (2012)		x			x		depot in the left corner of front cross aisle	x		x	
Chen et al. (2013)		x			x		depot in the left corner of front cross aisle	x		x	
Henn et al. (2013)			x		x		depot in the middle of front cross aisle	x		x	
Çelik and Süral (2014)			x		x		depot in the middle of front cross aisle	x		x	
Grosse et al. (2014)	x				x		depot in the middle of front cross aisle	x		x	

Papers	Warehouse layout										
	Conventional		Non-conventional	General	Number of depots		Depot location(s) (*=not mentioned in the paper)	Narrow aisle	Wide aisle	Low-level storage racks	High-level storage racks
	Single-block	Multi-block			One	Multiple					
Matusiak et al. (2014)		x				x	multiple depot locations	x		x	
Charkhgard and Savelsbergh (2015)	x				x		depot in the left corner of front cross aisle		x	x	
Chen et al. (2015)	x				x		depot in the left corner of front cross aisle	x		x	
Çelik and Süral (2016)	x				x		considered different depot locations	x		x	
Chen et al. (2016)		x			x		depot in the left corner of front cross aisle	x		x	
Lin et al. (2016)	x				x		depot in the left corner of front cross aisle	x		x	
Lu et al. (2016)	x				x		depot in the middle of front cross aisle	x		x	
Cortés et al. (2017)	x				x		depot in the left corner of front cross aisle	x		x	
Chabot et al. (2017)	x				x		depot in the middle of front cross aisle		x	x	
Li et al. (2017)		x			x		*	x		x	
Matusiak et al. (2017)		x				x	multiple depot locations	x		x	
Menéndez et al. (2017)	x				x		depot in the left corner of front cross aisle	x		x	
Scholz et al. (2017)		x			x		depot in the left corner of front cross aisle	x		x	
Scholz and Wäscher (2017)		x			x		depot in the left corner of front cross aisle	x		x	
Schrotenboer et al. (2017)	x				x		depot in the left corner of front cross aisle	x		x	
Ardjmand et al. (2018)	x				x		depot in the left corner of front cross aisle	x		x	
Bódis and Botzheim (2018)	x				x		depot in the left corner of back cross aisle	x		x	
De Santis et al. (2018)		x			x		depot in the left corner of front cross aisle	x		x	
Pansart et al. (2018)		x			x		depot in the middle of front cross aisle	x		x	
Pferschy and Schauer (2018)		x				x	multiple depot locations	x		x	
Weidinger (2018)	x				x		depot in the middle of front cross aisle	x		x	
Žulj et al. (2018)	x				x		depot in the left corner of front cross aisle	x		x	
Ardjmand et al. (2019)				x	x		*			x	
Çelik and Süral (2019)		x			x		depot in the left corner of front cross aisle	x		x	
Chen et al. (2019a)		x			x		depot in the left corner of front cross aisle	x		x	

Papers	Warehouse layout: Conventional – Single-block	Conventional – Multi-block	Non-conventional	General	Number of depots – One	Number of depots – Multiple	Depot location(s) (*=not mentioned in the paper)	Narrow aisle	Wide aisle	Low-level storage racks	High-level storage racks
Chen et al. (2019b)		x			x		depot in the left corner of front cross aisle	x		x	
Glock et al. (2019)			x		x		depot in the open end of the U-zone		x	x	
Öztürkoğlu and Hoser (2019)			x		x		depot in the left corner of front cross aisle	x		x	
Weidinger et al. (2019)	x					x	multiple depot locations	x		x	
Zhou et al. (2019)			x		x		depot in the middle of front cross aisle	x		x	
Frequency	**24**	**21**	**6**	**3**	**45**	**8**		**47**	**6**	**54**	**0**

Papers	Warehouse operations: Number of pickers – One	Number of pickers – Two	Number of pickers – Multiple	Start and end points of a picking tour	Capacity constraint	Picker congestion	Static picking	Dynamic picking	Pick-by-order	Pick-by-batch	Precedence constraints	Single storage	Scattered storage	Human factors	Algorithm characteristics: Exact	Heuristic	Meta-heuristic
Ratliff and Rosenthal (1983)	x			same location at the depot			x		x			x			x		
Goetschalckx and Ratliff (1988a)	x			different location			x		x			x			x	x	
Goetschalckx and Ratliff (1988b)	x			*			x		x			x			x		
Hall (1993)	x			same location at the depot			x		x			x				x	
Petersen (1997)	x			same location at the depot			x		x			x				x	
Singh and van Oudheusden (1997)	x			*			x						x		x		
Daniels et al. (1998)	x			*			x						x			x	x
De Koster and van der Poort (1998)	x			different location			x					x			x		
Vaughan and Petersen (1999)	x			different location			x		x			x				x	
Roodbergen and de Koster (2001a)	x			same location at the depot			x					x			x		
Roodbergen and de Koster (2001b)	x			same location at the depot			x					x				x	
Makris and Giakoumakis (2003)	x			different location			x					x				x	
Ho and Tseng (2006)	x			same location at the depot	x		x			x		x					x
Hsieh and Tsai (2006)	x			different location			x		x			x				x	

Papers	Warehouse operations														Algorithm characteristics		
	Number of pickers			Start and end points of a picking tour	Capacity constraint	Picker congestion	Static picking	Dynamic picking	Pick-by-order	Pick-by-batch	Precedence constraints	Single storage	Scattered storage	Human factors	Exact	Heuristic	Meta-heuristic
	One	Two	Multiple														
Shouman et al. (2007)	x			same location at the depot			x					x				x	
Tsai et al. (2008)	x			same location at the depot	x		x			x		x					x
Theys et al. (2010)	x			same location at the depot			x		x			x			x	x	
Glock and Grosse (2012)	x			same location at the depot	x		x		x			x				x	
Jang and Sun (2012)	x			same location at the depot			x					x			x		
Kulak et al. (2012)	x			same location at the depot	x		x			x		x				x	
Chen et al. (2013)		x		same location at the depot		x	x		x			x				x	x
Henn et al. (2013)	x			same location at the depot			x					x				x	
Çelik and Süral (2014)	x			same location at the depot			x		x			x			x	x	
Grosse et al. (2014)	x			same location at the depot	x		x			x		x				x	
Matusiak et al. (2014)	x			different location	x		x			x	x	x			x		
Charkhgard and Savelsbergh (2015)	x			different location			x		x			x			x		
Chen et al. (2015)	x			same location at the depot	x		x			x		x					x
Çelik and Süral (2016)	x			same location at the depot			x		x			x			x		
Chen et al. (2016)			x	same location at the depot		x	x		x			x					x
Lin et al. (2016)	x			same location at the depot	x		x			x		x					x
Lu et al. (2016)	x			different location				x				x			x		
Cortés et al. (2017)	x			same location at the depot	x		x			x		x					x
Chabot et al. (2017)	x			same location at the depot	x		x				x	x			x	x	x
Li et al. (2017)	x			same location at the depot	x		x			x		x					x
Matusiak et al. (2017)	x			different location	x		x			x	x	x				x	
Menéndez et al. (2017)	x			same location at the depot	x		x			x		x				x	
Scholz et al. (2017)			x	same location at the depot	x		x			x		x				x	
Scholz and Wäscher (2017)	x			same location at the depot	x		x			x		x				x	
Schrotenboer et al. (2017)			x	same location at the depot		x	x					x					x

Papers	Warehouse operations														Algorithm characteristics		
	Number of pickers			Start and end points of a picking tour	Capacity constraint	Picker congestion	Static picking	Dynamic picking	Pick-by-order	Pick-by-batch	Precedence constraints	Single storage	Scattered storage	Human factors	Exact	Heuristic	Meta-heuristic
	One	Two	Multiple														
Ardjmand et al. (2018)			x	same location at the depot			x			x		x					x
Bódis and Botzheim (2018)	x			same location at the depot	x		x					x					x
De Santis et al. (2018)	x			same location at the depot			x	x				x					x
Pansart et al. (2018)	x			same location at the depot			x					x			x		
Pferschy and Schauer (2018)	x			different location	x		x			x		x				x	
Weidinger (2018)	x			same location at the depot			x						x			x	
Žulj et al. (2018)	x			different location			x		x		x	x			x		
Ardjmand et al. (2019)	x			same location at the depot	x		x			x		x					x
Çelik and Süral (2019)	x			same location at the depot			x		x			x				x	
Chen et al. (2019a)	x			same location at the depot			x			x		x				x	
Chen et al. (2019b)	x			same location at the depot			x			x		x					x
Glock et al. (2019)	x			same location at the depot			x		x			x		x	x		
Öztürkoğlu and Hoser (2019)	x			same location at the depot			x			x		x			x		
Weidinger et al. (2019)	x			different location	x		x			x			x			x	
Zhou et al. (2019)	x			same location at the depot			x		x			x					x
Frequency	**49**	**1**	**4**		**19**	**3**	**53**	**2**	**18**	**20**	**4**	**50**	**4**	**1**	**18**	**26**	**17**

Table A.2 Average tour length per pick-list (meters) for the exact algorithms in combination with different storage assignment policies in chevron warehouse.

Storage assignment policies	Pick-list sizes	Average tour length per pick-list (average percentage gap from random storage) in the difference warehouse sizes								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
Random storage	1	16.02	23.06	31.17	38.49	45.18	52.87	61.16	68.21	75.71
	2	28.50	41.67	55.75	69.03	81.28	94.41	108.10	122.38	133.58
	3	38.43	56.56	75.81	92.98	109.41	128.35	147.25	164.80	181.43
	5	52.53	77.96	105.44	130.53	155.71	181.30	207.78	234.06	258.83
	10	70.63	110.92	154.03	193.52	233.01	274.89	313.14	354.01	395.15
	30	92.55	164.43	244.86	325.63	409.99	493.76	577.01	661.53	745.25
	60	101.18	183.07	286.47	401.18	522.87	650.07	781.47	919.05	1055.88
	Average	**57.12**	**93.95**	**136.22**	**178.77**	**222.49**	**267.95**	**313.70**	**360.58**	**406.55**
Turnover-based storage (20/40)	1	12.84 (19.85%)	18.04 (21.77%)	23.64 (24.16%)	29.33 (23.80%)	35.04 (22.44%)	40.82 (22.79%)	46.56 (23.87%)	52.31 (23.31%)	58.09 (23.27%)
	2	23.34 (18.11%)	33.66 (19.22%)	44.36 (20.43%)	54.91 (20.45%)	65.88 (18.95%)	76.29 (19.19%)	87.24 (19.30%)	98.94 (19.15%)	108.98 (18.42%)
	3	32.25 (16.08%)	46.86 (17.15%)	61.58 (18.77%)	76.51 (17.71%)	92.2 (15.73%)	106.98 (16.65%)	122.6 (16.74%)	137.83 (16.37%)	152.8 (15.78%)
	5	45.21 (13.93%)	67.41 (13.53%)	90.35 (14.31%)	112.86 (13.54%)	135.66 (12.88%)	158.89 (12.36%)	181.5 (12.65%)	203.22 (13.18%)	226.84 (12.36%)
	10	63.69 (9.83%)	100.76 (9.16%)	138.77 (9.91%)	177.47 (8.29%)	214.97 (7.74%)	250.06 (9.03%)	289.68 (7.49%)	327.02 (7.62%)	365.81 (7.43%)

Storage assignment policies	Pick-list sizes	Average tour length per pick-list (average percentage gap from random storage) in the difference warehouse sizes								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	30	88.18 (4.72%)	150.2 (8.65%)	224.37 (8.37%)	301.01 (7.56%)	381.23 (7.01%)	461.95 (6.44%)	544.28 (5.67%)	625.16 (5.50%)	705.05 (5.39%)
	60	100.32 (0.85%)	174.39 (4.74%)	266.86 (6.85%)	374.19 (6.73%)	488.02 (6.67%)	606.85 (6.65%)	730.78 (6.49%)	859.9 (6.44%)	988.82 (6.35%)
	Average	**52.26 (11.91%)**	**84.47 (13.46%)**	**121.42 (14.68%)**	**160.90 (14.01%)**	**201.86 (13.06%)**	**243.12 (13.30%)**	**286.09 (13.17%)**	**329.20 (13.08%)**	**372.34 (12.71%)**
Turnover-based storage (20/60)	1	10.72 (33.08%)	14.79 (35.86%)	19.15 (38.56%)	23.6 (38.69%)	28.11 (37.78%)	32.59 (38.36%)	37.11 (39.32%)	41.64 (38.95%)	46.18 (39.00%)
	2	19.04 (33.19%)	27.43 (34.17%)	36.18 (35.10%)	44.41 (35.67%)	53.42 (34.28%)	62.13 (34.19%)	70.52 (34.76%)	78.85 (35.57%)	87.67 (34.37%)
	3	26.21 (31.80%)	38.38 (32.14%)	50.64 (33.20%)	62.9 (32.35%)	74.57 (31.84%)	87.7 (31.67%)	100.07 (32.04%)	111.27 (32.48%)	125.06 (31.07%)
	5	37.41 (28.78%)	56.22 (27.89%)	75.75 (28.16%)	93.28 (28.54%)	112.9 (27.49%)	131.42 (27.51%)	150.83 (27.41%)	167.31 (28.52%)	187.95 (27.38%)
	10	55.37 (21.61%)	85.89 (22.57%)	117.65 (23.62%)	149.85 (22.57%)	182.72 (21.58%)	214.08 (22.12%)	246.4 (21.31%)	277.96 (21.48%)	310.11 (21.52%)
	30	85.2 (7.94%)	134.08 (18.46%)	195.7 (20.08%)	260.6 (19.97%)	327.83 (20.04%)	395.5 (19.90%)	463.13 (19.74%)	534.86 (19.15%)	606.89 (18.57%)
	60	99.89 (1.27%)	165.25 (9.73%)	244.03 (14.81%)	333.31 (16.92%)	429.69 (17.82%)	528.18 (18.75%)	632.21 (19.10%)	738.68 (19.63%)	848.78 (19.61%)
	Average	**47.69 (22.53%)**	**74.58 (25.83%)**	**105.59 (27.65%)**	**138.28 (27.81%)**	**172.75 (27.26%)**	**207.37 (27.50%)**	**242.90 (27.67%)**	**278.65 (27.97%)**	**316.09 (27.36%)**
Turnover-based storage (20/80)	1	8.67 (45.88%)	11.56 (49.87%)	14.66 (52.97%)	17.82 (53.70%)	21 (53.52%)	24.22 (54.19%)	27.52 (55.00%)	30.7 (54.99%)	33.94 (55.17%)
	2	15.28 (46.39%)	21.43 (48.57%)	27.7 (50.31%)	33.16 (51.96%)	39.68 (51.18%)	46.13 (51.14%)	52.42 (51.51%)	58.89 (51.88%)	64.65 (51.60%)

Storage assignment policies	Pick-list sizes	Average tour length per pick-list (average percentage gap from random storage) in the difference warehouse sizes								
		WZ1	WZ2	WZ3	WZ4	WZ5	WZ6	WZ7	WZ8	WZ9
	3	20.49 (46.68%)	29.87 (47.19%)	38.58 (49.11%)	47.73 (48.67%)	56.51 (48.35%)	65.17 (49.22%)	74.68 (49.28%)	83.48 (49.34%)	92.14 (49.21%)
	5	28.86 (45.06%)	44.13 (43.39%)	58.01 (44.98%)	72.27 (44.63%)	85.5 (45.09%)	99.54 (45.10%)	113.84 (45.21%)	127.16 (45.67%)	141.57 (45.30%)
	10	46.37 (34.35%)	68.43 (38.31%)	92.6 (39.88%)	116.36 (39.87%)	139.77 (40.02%)	164.65 (40.10%)	188.61 (39.77%)	212.86 (39.87%)	237.04 (40.01%)
	30	84.29 (8.92%)	118.91 (27.68%)	162.05 (33.82%)	208.4 (36.00%)	257.32 (37.24%)	307.46 (37.73%)	360.56 (37.51%)	413.08 (37.56%)	464.75 (37.64%)
	60	99.8 (1.36%)	157.13 (14.17%)	219.94 (23.22%)	285.87 (28.74%)	354.17 (32.26%)	427.94 (34.17%)	503.17 (35.61%)	581.91 (36.68%)	663.6 (37.15%)
	Average	**43.39 (32.66%)**	**64.49 (38.45%)**	**87.65 (42.04%)**	**111.66 (43.37%)**	**136.28 (43.95%)**	**162.16 (44.52%)**	**188.69 (44.84%)**	**215.44 (45.14%)**	**242.53 (45.16%)**

Table A.3 Percentage optimality gaps of the heuristics for the case of a random storage policy in chevron warehouse.

Routing heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using random storage policy								
		WZ1	WZ2	WZ3	WZ4	WZ5	WZ6	WZ7	WZ8	WZ9
CM	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	3.33%	5.42%	4.97%	4.74%	5.71%	5.23%	5.38%	5.45%	5.84%
	3	4.92%	7.50%	7.35%	8.02%	9.56%	8.49%	8.98%	9.17%	8.74%

Routing heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using random storage policy								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	5	7.79%	10.24%	10.76%	11.25%	12.48%	10.88%	12.50%	11.43%	11.84%
	10	16.61%	15.57%	14.08%	14.26%	13.95%	13.12%	13.33%	13.34%	13.07%
	30	29.61%	29.34%	24.14%	21.27%	18.95%	17.09%	15.77%	14.44%	14.05%
	60	22.51%	44.01%	38.09%	33.58%	29.38%	25.96%	23.33%	21.14%	19.50%
	Average	**12.11%**	**16.01%**	**14.20%**	**13.30%**	**12.86%**	**11.54%**	**11.33%**	**10.71%**	**10.43%**
CL	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	3.33%	5.42%	4.97%	4.74%	5.71%	5.23%	5.38%	5.45%	5.84%
	3	4.92%	7.50%	7.35%	8.02%	9.56%	8.49%	8.98%	9.17%	8.74%
	5	7.60%	9.97%	10.57%	11.15%	12.43%	10.79%	12.41%	11.38%	11.74%
	10	13.62%	13.61%	12.71%	13.10%	12.93%	12.38%	12.52%	12.66%	12.50%
	30	21.25%	21.44%	17.51%	15.48%	13.74%	12.52%	11.59%	10.87%	10.42%
	60	14.63%	34.50%	28.12%	23.88%	20.07%	17.43%	15.33%	13.86%	12.48%

Routing heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using random storage policy								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	Average	**9.34%**	**13.21%**	**11.60%**	**10.91%**	**10.63%**	**9.55%**	**9.46%**	**9.06%**	**8.82%**
CS	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	1.79%	3.17%	2.91%	2.67%	3.17%	3.52%	3.16%	3.21%	3.38%
	3	3.96%	6.68%	6.71%	7.92%	9.27%	9.26%	9.35%	9.59%	9.34%
	5	5.75%	10.83%	13.88%	14.96%	16.38%	16.95%	18.02%	17.10%	18.32%
	10	7.52%	15.26%	20.22%	22.49%	25.03%	26.77%	28.55%	29.44%	30.11%
	30	5.05%	5.80%	11.07%	16.34%	20.74%	24.29%	27.81%	30.79%	33.26%
	60	0.53%	3.17%	3.65%	6.53%	9.89%	13.00%	16.36%	19.14%	22.27%
	Average	**3.51%**	**6.42%**	**8.35%**	**10.13%**	**12.07%**	**13.40%**	**14.75%**	**15.61%**	**16.67%**

Table A.4 Percentage optimality gaps of the heuristics for the case of a turnover-based storage policy with a 20/40 demand skewness in chevron warehouse.

Routing heuristics	**Pick-list sizes**	**Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/40)**								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
CM	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	6.26%	10.22%	12.67%	12.73%	13.43%	13.97%	13.18%	12.95%	13.78%
	3	8.09%	16.18%	17.72%	19.44%	20.02%	19.65%	19.24%	20.64%	20.21%
	5	11.37%	20.13%	21.45%	22.61%	22.29%	22.36%	22.48%	23.21%	23.21%
	10	15.56%	21.11%	20.11%	19.95%	19.61%	19.48%	18.62%	19.08%	19.02%
	30	34.83%	30.15%	26.95%	23.51%	21.03%	19.07%	18.06%	16.73%	15.83%
	60	23.60%	43.71%	38.45%	33.65%	29.97%	26.95%	24.82%	22.53%	20.66%
	Average	**14.24%**	**20.21%**	**19.62%**	**18.84%**	**18.05%**	**17.35%**	**16.63%**	**16.45%**	**16.10%**
CL	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	6.26%	10.22%	12.67%	12.73%	13.43%	13.97%	13.18%	12.95%	13.78%

Routing heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/40)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	3	8.09%	16.18%	17.72%	19.44%	20.02%	19.65%	19.24%	20.64%	20.21%
	5	11.19%	19.97%	21.31%	22.28%	22.14%	22.27%	22.41%	23.07%	23.14%
	10	13.27%	19.16%	18.63%	18.48%	18.50%	18.62%	17.87%	18.31%	18.35%
	30	24.79%	21.37%	19.31%	17.06%	15.52%	14.29%	13.61%	12.80%	12.22%
	60	15.54%	30.62%	25.82%	22.18%	19.73%	17.66%	16.13%	14.58%	13.34%
	Average	**11.31%**	**16.79%**	**16.49%**	**16.02%**	**15.62%**	**15.21%**	**14.64%**	**14.62%**	**14.43%**
CS	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	5.10%	7.81%	10.03%	10.13%	11.38%	12.39%	10.81%	10.01%	11.86%
	3	7.72%	14.23%	16.03%	18.51%	19.26%	18.91%	18.65%	19.20%	19.98%
	5	9.87%	18.26%	21.73%	24.34%	25.15%	25.68%	26.64%	27.88%	28.37%
	10	7.40%	17.27%	23.60%	26.82%	30.25%	32.92%	34.02%	35.95%	37.45%

Routing heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/40)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	30	5.08%	7.90%	13.17%	18.96%	23.32%	27.47%	30.98%	33.99%	36.84%
	60	0.78%	2.98%	5.52%	8.98%	12.39%	15.96%	18.99%	22.19%	25.23%
	Average	**5.13%**	**9.78%**	**12.87%**	**15.39%**	**17.39%**	**19.05%**	**20.01%**	**21.32%**	**22.82%**

Table A.5 Percentage optimality gaps of the heuristics for the case of a turnover-based storage policy with a 20/60 demand skewness in chevron warehouse.

Routing heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/60)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
CM	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	7.72%	17.13%	20.78%	19.73%	22.93%	24.59%	24.11%	25.19%	25.27%
	3	11.64%	26.60%	29.90%	32.26%	33.43%	34.39%	35.38%	36.84%	35.18%

Routing heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/60)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	5	14.17%	30.79%	34.59%	37.53%	36.77%	38.03%	38.55%	39.29%	38.41%
	10	16.18%	28.89%	28.46%	29.39%	28.19%	28.46%	28.72%	28.51%	28.26%
	30	37.49%	28.57%	25.97%	23.56%	21.53%	20.11%	18.97%	18.14%	17.91%
	60	24.14%	39.27%	33.81%	29.57%	26.01%	23.68%	21.85%	20.00%	19.15%
	Average	**15.90%**	**24.46%**	**24.79%**	**24.58%**	**24.12%**	**24.18%**	**23.94%**	**23.99%**	**23.45%**
CL	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	7.72%	17.13%	20.78%	19.73%	22.93%	24.59%	24.11%	25.19%	25.27%
	3	11.64%	26.60%	29.90%	32.26%	33.43%	34.39%	35.38%	36.84%	35.18%
	5	14.09%	30.58%	34.52%	37.45%	36.72%	37.94%	38.47%	39.20%	38.36%
	10	14.70%	27.50%	27.29%	28.48%	27.49%	27.84%	28.26%	28.03%	27.62%
	30	26.29%	20.81%	19.99%	18.70%	17.46%	16.54%	15.81%	15.33%	14.96%

Routing heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/60)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	60	16.02%	25.89%	22.38%	19.88%	17.81%	16.19%	15.03%	13.85%	13.36%
	Average	**12.92%**	**21.22%**	**22.12%**	**22.36%**	**22.26%**	**22.50%**	**22.44%**	**22.63%**	**22.11%**
CS	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	7.09%	15.20%	19.18%	17.09%	20.89%	22.87%	22.49%	22.99%	23.38%
	3	11.18%	24.78%	28.77%	31.07%	32.41%	34.47%	34.47%	35.97%	35.66%
	5	13.85%	29.49%	35.93%	39.67%	40.65%	42.92%	43.87%	46.00%	44.57%
	10	10.69%	25.22%	33.50%	39.73%	43.45%	47.17%	51.27%	52.55%	54.11%
	30	5.53%	11.67%	19.31%	26.31%	32.10%	37.33%	43.04%	46.25%	48.93%
	60	0.92%	4.04%	9.24%	14.72%	19.18%	23.91%	27.58%	31.80%	35.16%
	Average	**7.04%**	**15.77%**	**20.85%**	**24.08%**	**26.95%**	**29.81%**	**31.82%**	**33.65%**	**34.55%**

Table A.6 Percentage optimality gaps of the heuristics for the case of a turnover-based storage policy with a 20/80 demand skewness in chevron warehouse.

Routing heuristics	**Pick-list sizes**	**Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/80)**								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
CM	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	6.48%	22.40%	30.69%	31.54%	41.76%	39.56%	44.01%	43.34%	46.51%
	3	11.91%	34.38%	45.80%	51.54%	57.92%	61.21%	62.41%	62.59%	64.99%
	5	16.67%	42.53%	55.80%	60.27%	63.26%	67.94%	66.48%	68.91%	70.77%
	10	16.95%	38.70%	45.14%	46.80%	47.89%	48.35%	48.64%	49.37%	49.81%
	30	38.60%	29.71%	27.76%	25.91%	25.15%	24.77%	24.00%	23.00%	22.73%
	60	24.25%	36.10%	30.05%	25.62%	23.29%	21.42%	20.42%	18.95%	18.46%
	Average	**16.41%**	**29.12%**	**33.61%**	**34.53%**	**37.04%**	**37.61%**	**37.99%**	**38.02%**	**39.04%**
CL	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	6.48%	22.40%	30.69%	31.54%	41.76%	39.56%	44.01%	43.34%	46.51%

Routing heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/80)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	3	11.91%	34.38%	45.80%	51.54%	57.92%	61.21%	62.41%	62.59%	64.99%
	5	16.60%	42.53%	55.71%	60.19%	63.22%	67.87%	66.48%	68.83%	70.71%
	10	16.50%	38.16%	44.63%	46.53%	47.65%	48.00%	48.41%	49.07%	49.61%
	30	26.39%	23.80%	24.01%	23.44%	23.03%	22.78%	22.24%	21.63%	21.40%
	60	16.13%	23.14%	20.89%	19.15%	17.98%	16.87%	16.46%	15.57%	15.16%
	Average	**13.43%**	**26.34%**	**31.68%**	**33.20%**	**35.94%**	**36.61%**	**37.14%**	**37.29%**	**38.34%**
CS	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	6.35%	20.77%	29.13%	29.46%	39.87%	37.70%	41.91%	41.28%	44.24%
	3	11.86%	33.95%	45.02%	50.41%	57.16%	60.23%	61.92%	62.84%	64.16%
	5	16.63%	42.19%	55.94%	62.02%	67.68%	72.19%	73.83%	76.00%	79.19%
	10	14.56%	35.89%	50.19%	58.03%	66.27%	71.78%	76.87%	80.88%	85.86%

Routing heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/80)								
		WZ1	WZ2	WZ3	WZ4	WZ5	WZ6	WZ7	WZ8	WZ9
	30	5.30%	16.00%	29.58%	38.45%	47.35%	54.08%	59.08%	65.79%	71.68%
	60	0.99%	5.53%	14.04%	22.71%	30.03%	36.08%	42.09%	46.86%	52.27%
	Average	**7.96%**	**22.05%**	**31.99%**	**37.30%**	**44.05%**	**47.44%**	**50.81%**	**53.38%**	**56.77%**

Table A.7 Percentage gaps between the average tour length in conventional two-block and chevron warehouses under different storage assignment policies.

Storage assignment policies	Pick-list sizes	Percentage gaps between the average tour length in conventional two-block and chevron warehouses								
		WZ1	WZ2	WZ3	WZ4	WZ5	WZ6	WZ7	WZ8	WZ9
Random storage	1	30.56%	10.92%	5.23%	2.86%	-3.97%	-5.17%	-3.35%	-6.99%	-5.48%
	2	37.28%	21.95%	13.08%	13.59%	6.83%	5.62%	5.11%	4.36%	4.03%
	3	43.18%	29.81%	20.09%	19.02%	11.91%	12.87%	12.84%	10.72%	9.07%
	5	54.18%	35.32%	24.68%	27.66%	19.70%	19.35%	19.22%	18.11%	17.59%

Storage assignment policies	Pick-list sizes	Percentage gaps between the average tour length in conventional two-block and chevron warehouses								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	10	74.27%	42.11%	29.82%	31.54%	25.36%	24.72%	22.43%	22.65%	22.65%
	30	120.36%	50.78%	35.30%	32.65%	25.71%	23.42%	21.06%	20.20%	19.17%
	60	140.90%	60.64%	39.18%	33.94%	25.63%	21.19%	19.48%	17.64%	16.75%
	Average	**71.53%**	**35.94%**	**23.91%**	**23.04%**	**15.88%**	**14.57%**	**13.83%**	**12.38%**	**11.97%**
Turnover-based storage (20/40)	1	43.78%	13.82%	3.91%	1.35%	-2.34%	-4.63%	-4.96%	-6.42%	-6.37%
	2	45.60%	21.47%	11.82%	10.71%	7.63%	4.89%	5.12%	4.58%	3.23%
	3	50.00%	26.72%	15.93%	17.51%	13.63%	10.83%	12.55%	10.55%	10.67%
	5	56.11%	33.70%	23.33%	26.10%	20.34%	19.21%	19.75%	17.96%	17.44%
	10	68.18%	40.24%	30.36%	32.46%	26.35%	24.50%	23.59%	23.67%	22.94%
	30	110.00%	45.19%	33.60%	32.19%	26.22%	24.75%	24.34%	23.40%	21.60%
	60	138.86%	53.74%	35.43%	32.47%	25.17%	21.99%	20.70%	19.77%	18.69%
	Average	**73.22%**	**33.55%**	**22.06%**	**21.83%**	**16.71%**	**14.51%**	**14.44%**	**13.36%**	**12.60%**

Storage assignment policies	Pick-list sizes	Percentage gaps between the average tour length in conventional two-block and chevron warehouses								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
Turnover-based storage (20/60)	1	58.58%	23.04%	10.44%	6.69%	2.14%	-0.91%	-1.54%	-3.25%	-3.57%
	2	55.68%	26.29%	16.52%	12.80%	9.74%	7.14%	6.93%	4.58%	4.64%
	3	56.20%	29.88%	20.29%	19.04%	12.88%	12.44%	12.60%	9.69%	11.54%
	5	56.72%	33.95%	25.37%	23.76%	20.17%	18.26%	18.13%	15.27%	16.86%
	10	64.64%	40.94%	29.39%	31.03%	24.62%	23.77%	23.35%	22.27%	21.24%
	30	103.15%	43.19%	31.73%	33.11%	26.26%	25.06%	23.13%	22.68%	21.44%
	60	137.83%	48.69%	32.95%	31.82%	25.21%	22.96%	21.43%	20.05%	19.03%
	Average	**76.12%**	**35.14%**	**23.81%**	**22.61%**	**17.29%**	**15.53%**	**14.86%**	**13.04%**	**13.02%**
Turnover-based storage (20/80)	1	84.86%	39.61%	21.86%	15.49%	9.15%	4.89%	3.73%	1.12%	0.38%
	2	77.67%	40.52%	25.97%	18.01%	14.38%	10.86%	10.27%	8.57%	7.52%
	3	70.89%	40.30%	27.16%	23.27%	17.36%	14.11%	15.14%	12.58%	11.89%
	5	61.59%	41.35%	31.60%	29.03%	21.62%	19.00%	19.91%	16.82%	16.55%

Storage assignment policies	**Pick-list sizes**	**Percentage gaps between the average tour length in conventional two-block and chevron warehouses**								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	10	60.95%	43.61%	33.89%	32.27%	24.61%	22.70%	22.30%	21.56%	20.35%
	30	101.31%	43.63%	31.77%	32.84%	26.68%	24.90%	24.02%	23.29%	20.94%
	60	137.62%	45.96%	32.99%	32.97%	25.62%	23.59%	22.08%	21.10%	20.06%
	Average	**84.99%**	**42.14%**	**29.32%**	**26.27%**	**19.92%**	**17.15%**	**16.78%**	**15.01%**	**13.95%**

Table A.8 Equivalence classes of $L_{n_t}^{12}$ ($L_{n_t}^{34}$) PTSs that result from the union of all equivalence classes of $L_{n_t,2}^{+u}$ ($L_{n_t,3}^{+u}$) and $L_{n_s,1}^{+}$ ($L_{n_s,4}^{+}$) PTSs of leaf warehouse.

Equivalence classes of $L_{n_t,2}^{+u}$ ($L_{n_t,3}^{+u}$) PTSs	Equivalence classes of $L_{n_s,1}^{+}$ ($L_{n_s,4}^{+u}$) PTSs						
	$(U,U,1C)$	$(E,E,1C)$	$(E,0,1C)$	$(0,E,1C)$	$(E,E,2C)$	$(0,0,0C)$	$(0,0,1C)$
$(0,0,0,0C)$	$(0,U,U,1C)$	$(0,E,E,1C)$	$(0,E,0,1C)$	$(0,0,E,1C)$	$(0,E,E,2C)$	$(0,0,0,0C)$	NA
$(0,0,0,1C)$	NA	NA	NA	NA	NA	$(0,0,0,0,1C)$	NA
$(E,0,0,1C)$	$(E,U,U,2C)$	$(E,E,E,2C,a-bc)$	$(E,E,0,2C)$	$(E,0,E,2C)$	$(E,E,E,3C)$	$(E,0,0,1C)$	NA
$(0,E,0,1C)$	$(0,U,U,1C)$	$(0,E,E,1C)$	$(0,E,0,1C)$	$(0,E,E,2C)$	$(0,E,E,2C)$	$(0,E,0,1C)$	NA
$(0,0,E,1C)$	$(O,U,U,1C)$	$(0,E,E,1C)$	$(0,E,E,2C)$	$(0,0,E,1C)$	$(0,E,E,2C)$	$(0,0,E,1C)$	NA
$(E,E,0,1C)$	$(E,U,U,1C)$	$(E,E,E,1C)$	$(E,E,0,1C)$	$(E,E,E,2C,c-ab)$	$(E,E,E,2C,c-ab)$	$(E,E,0,1C)$	NA
$(E,0,E,1C)$	$(E,U,U,1C)$	$(E,E,E,1C)$	$(E,E,E,2C,b-ac)$	$(E,0,E,1C)$	$(E,E,E,2C,b-ac)$	$(E,0,E,1C)$	NA
$(0,E,E,1C)$	$(0,U,U,1C)$	NA	$(0,E,E,1C)$	$(0,E,E,1C)$	$(0,E,E,1C)$	$(0,E,E,1C)$	NA
$(E,E,E,1C)$	$(E,U,U,1C)$	NA	$(E,E,E,1C)$	$(E,E,E,1C)$	$(E,E,E,1C)$	$(E,E,E,1C)$	NA

Equivalence classes of $L^{+u}_{n_t,2}$ ($L^{+u}_{n_t,3}$) PTSs	Equivalence classes of $L^{+}_{n_s,1}$ ($L^{+u}_{n_s,4}$) PTSs						
	$(U,U,1C)$	$(E,E,1C)$	$(E,0,1C)$	$(0,E,1C)$	$(E,E,2C)$	$(0,0,0C)$	$(0,0,1C)$
$(U,U,0,1C)$	$(U,E,U,1C)$	$(U,U,E,1C)$	$(U,U,0,1C)$	$(U,U,E,2C)$	$(U,U,E,2C)$	$(U,U,0,1C)$	NA
$(U,0,U,1C)$	$(U,U,E,1C)$	$(U,E,U,1C)$	$(U,E,U,2C)$	$(U,0,U,1C)$	$(U,E,U,2C)$	$(U,0,U,1C)$	NA
$(0,U,U,1C)$	$(0,E,E,1C)$	NA	$(0,U,U,1C)$	$(0,U,U,1C)$	$(0,U,U,1C)$	$(0,U,U,1C)$	NA
$(E,U,U,1C)$	$(E,E,E,1C)$	NA	$(E,U,U,1C)$	$(E,U,U,1C)$	$(E,U,U,1C)$	$(E,U,U,1C)$	NA
$(U,E,U,1C)$	$(U,U,E,1C)$	NA	$(U,E,U,1C)$	$(U,E,U,1C)$	$(U,E,U,1C)$	$(U,E,U,1C)$	NA
$(U,U,E,1C)$	$(U,E,U,1C)$	NA	$(U,U,E,1C)$	$(U,U,E,1C)$	$(U,U,E,1C)$	$(U,U,E,1C)$	NA
$(E,E,0,2C)$	$(E,U,U,2C)$	$(E,E,E,2C,a-bc)$	$(E,E,0,2C)$	$(E,E,E,3C)$	$(E,E,E,3C)$	$(E,E,0,2C)$	NA
$(E,0,E,2C)$	$(E,U,U,2C)$	$(E,E,E,2C,a-bc)$	$(E,E,E,3C)$	$(E,0,E,2C)$	$(E,E,E,3C)$	$(E,0,E,2C)$	NA
$(0,E,E,2C)$	$(0,U,U,1C)$	$(0,E,E,1)$	$(0,E,E,2C)$	$(0,E,E,2C)$	$(0,E,E,2C)$	$(0,E,E,2C)$	NA
$(E,E,E,2C,a-bc)$	$(E,U,U,2C)$	NA	$(E,E,E,2C,a-bc)$	$(E,E,E,2C,a-bc)$	$(E,E,E,2C,a-bc)$	$(E,E,E,2C,a-bc)$	NA
$(E,E,E,2C,b-ac)$	$(E,U,U,1C)$	$(E,E,E,1C)$	$(E,E,E,2C,b-ac)$	$(E,E,E,2C,b-ac)$	$(E,E,E,2C,b-ac)$	$(E,E,E,2C,b-ac)$	NA
$(E,E,E,2C,c-ab)$	$(E,U,U,1C)$	$(E,E,E,1C)$	$(E,E,E,2C,c-ab)$	$(E,E,E,2C,c-ab)$	$(E,E,E,2C,c-ab)$	$(E,E,E,2C,c-ab)$	NA
$(E,U,U,2C)$	$(E,E,E,2C,a-bc)$	NA	$(E,U,U,2C)$	$(E,U,U,2C)$	$(E,U,U,2C)$	$(E,U,U,2C)$	NA
$(U,E,U,2C)$	$(U,U,E,1C)$	$(U,E,U,1C)$	$(U,E,U,2C)$	$(U,E,U,2C)$	$(U,E,U,2C)$	$(U,E,U,2C)$	NA
$(U,U,E,2C)$	$(U,E,U,1C)$	$(U,U,E,1C)$	$(U,U,E,2C)$	$(U,U,E,2C)$	$(U,U,E,2C)$	$(U,U,E,2C)$	NA
$(E,E,E,3C)$	$(E,U,U,2C)$	$(E,E,E,2C,a-bc)$	$(E,E,E,3C)$	$(E,E,E,3C)$	$(E,E,E,3C)$	$(E,E,E,3C)$	NA

Table A.9 Equivalence classes that result from the union of equivalence classes of $L^{12}_{n_t}$ and $L^{34}_{n_t}$ PTSs of leaf warehouse.

Equivalence classes of $L^{12}_{n_t}$	Equivalence classes of $L^{34}_{n_t}$													
	$(0,0,0,0C)$	$(0,0,0,1C)$	$(U,U,0,1C)$	$(E,0,0,1C)$	$(E,E,0,1C)$	$(E,E,0,2C)$	$(0,E,0,1C)$	$(0,E,E,1C)$	$(0,0,E,1C)$	$(E,0,E,1C)$	$(E,E,E,1C)$	$(U,U,E,1C)$	$(E,E,E,2C,a-bc)$	$(E,E,E,2C,b-ac)$
$(0,0,0,0C)$	$(0,0,0,0,0C)$													
$(0,0,0,1C)$	$(0,0,0,0,1C)$	NA												
$(U,U,0,1C)$	$(U,U,0,0,1C)$	NA	$(E,E,0,0,1C)$											
$(E,0,0,1C)$	$(E,0,0,0,1C)$	NA	$(U,U,0,0,1C)$	$(E,0,0,0,1C)$										

Equivalence classes of $L_{n_t}^{12}$	Equivalence classes of $L_{n_t}^{34}$													
	$(0,0,0,0C)$	$(0,0,0,1C)$	$(U,U,0,1C)$	$(E,0,0,1C)$	$(E,E,0,1C)$	$(E,E,0,2C)$	$(0,E,0,1C)$	$(0,E,E,1C)$	$(0,0,E,1C)$	$(E,0,E,1C)$	$(E,E,E,1C)$	$(U,U,E,1C)$	$(E,E,E,2C,a-bc)$	$(E,E,E,2C,b-ac)$
$(E,E,0,1C)$	$(E,E,0,0,1C)$	NA	$(U,U,0,0,1C)$	$(E,E,0,0,1C)$	$(E,E,0,0,1C)$									
$(E,E,0,2C)$	$(E,E,0,0,2C)$	NA	$(U,U,0,0,1C)$	$(E,E,0,0,2C)$	$(E,E,0,0,1C)$	$(E,E,0,0,2C)$								
$(0,E,0,1C)$	$(0,E,0,0,1C)$	NA	$(U,U,0,0,1C)$	$(E,E,0,0,2C)$	$(E,E,0,0,1C)$	$(E,E,0,0,2C)$	$(0,E,0,0,1C)$							
$(0,E,E,1C)$	$(0,E,E,0,1C)$	NA	$(U,U,E,0,1C)$	$(E,E,E,0,2C,a-bc)$	$(E,E,E,0,1C)$	$(E,E,E,0,2C,a-bc)$	$(0,E,E,0,1C)$	$(0,E,E,E,1C)$						
$(0,0,E,1C)$	$(0,0,E,0,1C)$	NA	$(U,U,E,0,2C)$	$(E,0,E,0,2C)$	$(E,E,E,0,2C,c-ab)$	$(E,E,E,0,3C)$	$(0,E,E,0,2C)$	$(0,E,E,E,2C,c-bc)$	$(0,0,E,E,2C)$					
$(E,0,E,1C)$	$(E,0,E,0,1C)$	NA	$(U,U,E,0,1C)$	$(E,0,E,0,1C)$	$(E,E,E,0,1C)$	$(E,E,E,0,2C,b-ac)$	$(E,E,E,0,2C,b-ac)$	$(E,E,E,E,2C,ac-bc)$	$(E,0,E,E,2C,ac-c)$	$(E,0,E,E,1C)$				
$(E,E,E,1C)$	$(E,E,E,0,1C)$	NA	$(U,U,E,0,1C)$	$(E,E,E,0,1C)$	$(E,E,E,0,1C)$	$(E,E,E,0,1C)$	$(E,E,E,0,1C)$	$(E,E,E,E,1C)$	$(E,E,E,E,2C,abc-c)$	$(E,E,E,E,1C)$	$(E,E,E,E,1C)$			
$(U,U,E,1C)$	$(U,U,E,0,1C)$	NA	$(E,E,E,0,1C)$	$(U,U,E,0,1C)$	$(U,U,E,0,1C)$	$(U,U,E,0,1C)$	$(U,U,E,0,1C)$	$(U,U,E,E,1C)$	$(U,U,E,E,2C,abc-c)$	$(U,U,E,E,1C)$	$(U,U,E,E,1C)$	$(E,E,E,E,1C)$		
$(E,E,E,2C,a-bc)$	$(E,E,E,0,2C,a-bc)$	NA	$(U,U,E,0,1C)$	$(E,E,E,0,2C,a-bc)$	$(E,E,E,0,1C)$	$(E,E,E,0,2C,a-bc)$	$(E,E,E,0,2C,a-bc)$	$(E,E,E,E,2C,a-bcc)$	$(E,E,E,E,3C,a-bc-c)$	$(E,E,E,E,2C,ac-bc)$	$(E,E,E,E,1C)$	$(U,U,E,E,1C)$	$(E,E,E,E,2C,a-bcc)$	
$(E,E,E,2C,b-ac)$	$(E,E,E,0,2C,b-ac)$	NA	$(U,U,E,0,1C)$	$(E,E,E,0,2C,b-ac)$	$(E,E,E,0,1C)$	$(E,E,E,0,2C,b-ac)$	$(E,E,E,0,2C,b-ac)$	$(E,E,E,E,2C,ac-bc)$	$(E,E,E,E,3C,ac-b-c)$	$(E,E,E,E,2C,acc-b)$	$(E,E,E,E,1C)$	$(U,U,E,E,1C)$	$(E,E,E,E,2C,ac-bc)$	$(E,E,E,E,2C,acc-b)$

Table A. 10 Percentage optimality gaps of the heuristics for the case of a random storage policy in leaf warehouse.

Heuristics	**Pick-list sizes**	**Percentage optimality gaps of the heuristics in different warehouse sizes using random storage policy**								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
LM	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	17.13%	15.90%	14.45%	13.16%	13.15%	14.74%	12.61%	14.63%	13.76%
	3	30.86%	28.70%	24.36%	24.12%	23.39%	23.05%	22.91%	22.74%	22.46%

Heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using random storage policy								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	5	45.66%	40.63%	34.19%	32.74%	33.10%	32.26%	32.04%	30.96%	30.90%
	10	58.29%	48.67%	40.97%	39.06%	36.87%	34.73%	34.94%	33.67%	33.11%
	30	73.11%	53.56%	41.13%	34.83%	30.82%	27.82%	25.50%	24.26%	22.50%
	60	80.71%	65.38%	51.28%	42.29%	35.64%	30.65%	27.63%	24.61%	22.63%
	Average	**43.68%**	**36.12%**	**29.48%**	**26.60%**	**24.71%**	**23.32%**	**22.23%**	**21.55%**	**20.77%**
LL	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	17.63%	15.39%	14.57%	13.37%	13.73%	14.99%	13.13%	14.54%	13.66%
	3	30.49%	27.92%	23.99%	24.11%	23.73%	23.45%	23.08%	22.32%	22.58%
	5	41.57%	36.97%	32.20%	31.65%	32.45%	31.98%	31.99%	30.29%	30.30%
	10	48.21%	42.81%	37.15%	36.17%	34.54%	33.17%	33.25%	32.51%	32.14%
	30	54.81%	41.64%	33.10%	28.34%	25.58%	23.30%	21.82%	21.02%	19.67%
	60	58.52%	47.42%	37.83%	31.07%	26.10%	22.41%	20.18%	18.19%	16.84%

Heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using random storage policy								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	Average	**35.89%**	**30.31%**	**25.55%**	**23.53%**	**22.30%**	**21.33%**	**20.49%**	**19.84%**	**19.31%**
LS	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	0.61%	0.66%	1.29%	1.54%	2.06%	1.83%	1.73%	2.12%	2.22%
	3	1.53%	2.39%	3.19%	3.99%	4.85%	4.23%	5.05%	5.68%	6.03%
	5	4.10%	6.74%	8.66%	10.14%	11.43%	11.91%	12.22%	13.00%	13.40%
	10	9.37%	15.71%	18.99%	21.77%	23.44%	23.97%	25.35%	26.99%	27.30%
	30	15.10%	14.95%	19.50%	23.54%	27.37%	29.68%	32.30%	35.30%	37.65%
	60	15.16%	9.30%	11.95%	14.77%	17.90%	20.93%	24.15%	27.32%	29.93%
	Average	**6.55%**	**7.11%**	**9.08%**	**10.82%**	**12.43%**	**13.22%**	**14.40%**	**15.77%**	**16.65%**
LR	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	5.00%	6.13%	5.91%	6.44%	5.68%	5.67%	5.82%	6.06%	6.11%
	3	9.90%	11.83%	11.61%	11.61%	12.17%	11.49%	11.45%	11.70%	12.08%

Heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using random storage policy								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	5	19.88%	22.23%	22.82%	23.37%	23.04%	22.54%	23.09%	23.65%	23.01%
	10	36.97%	40.61%	40.32%	40.31%	41.80%	40.39%	41.00%	41.27%	41.09%
	30	54.05%	55.37%	55.72%	56.99%	57.75%	56.73%	57.57%	59.26%	59.78%
	60	57.96%	59.18%	59.85%	58.93%	59.51%	59.78%	60.04%	61.45%	62.71%
	Average	**26.25%**	**27.91%**	**28.03%**	**28.24%**	**28.56%**	**28.09%**	**28.42%**	**29.06%**	**29.26%**

Table A.11 Percentage optimality gaps of the heuristics for the case of a turnover-based storage policy with a 20/40 demand skewness in leaf warehouse.

Heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/40)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
LM	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	16.39%	13.13%	13.41%	12.39%	13.20%	11.92%	12.41%	14.61%	13.46%
	3	25.40%	23.84%	21.43%	21.26%	20.90%	20.23%	20.90%	23.45%	21.60%

Heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/40)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	5	43.47%	37.10%	32.82%	31.81%	31.74%	31.13%	30.63%	33.48%	31.45%
	10	62.84%	50.93%	41.49%	38.80%	37.08%	36.35%	35.92%	35.36%	34.23%
	30	73.54%	57.36%	44.33%	37.08%	33.16%	29.59%	27.59%	26.06%	24.46%
	60	80.39%	65.71%	53.37%	44.10%	37.92%	32.85%	29.48%	26.76%	24.50%
	Average	**43.15%**	**35.44%**	**29.55%**	**26.49%**	**24.86%**	**23.15%**	**22.42%**	**22.82%**	**21.38%**
LL	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	15.82%	12.62%	13.39%	12.68%	13.29%	12.22%	12.59%	14.57%	13.37%
	3	23.92%	22.30%	20.78%	21.53%	20.63%	19.88%	20.91%	23.18%	21.27%
	5	37.06%	32.91%	30.59%	30.88%	31.14%	30.28%	30.62%	32.96%	30.87%
	10	47.66%	42.92%	37.34%	35.59%	34.98%	34.32%	34.06%	33.86%	33.23%
	30	53.19%	44.12%	35.31%	30.42%	27.52%	25.02%	23.46%	22.52%	21.28%
	60	58.51%	48.54%	39.76%	33.01%	28.07%	24.28%	21.79%	19.81%	18.07%

Heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/40)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	Average	**33.74%**	**29.06%**	**25.31%**	**23.44%**	**22.23%**	**20.86%**	**20.49%**	**20.99%**	**19.73%**
LS	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	0.40%	1.01%	2.15%	2.30%	2.93%	3.28%	3.73%	4.38%	4.30%
	3	0.86%	2.66%	3.98%	4.97%	6.17%	6.43%	6.97%	8.40%	7.76%
	5	2.37%	5.45%	8.29%	10.47%	11.88%	13.46%	13.97%	14.54%	15.37%
	10	5.56%	10.94%	15.19%	18.25%	20.83%	22.95%	23.93%	25.71%	26.90%
	30	11.07%	11.76%	15.60%	20.08%	23.79%	27.16%	30.18%	33.91%	35.30%
	60	15.10%	8.12%	11.27%	13.46%	16.15%	19.20%	22.38%	25.81%	28.42%
	Average	**5.05%**	**5.71%**	**8.07%**	**9.93%**	**11.68%**	**13.21%**	**14.45%**	**16.11%**	**16.86%**
LR	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	3.09%	3.11%	2.68%	2.50%	3.06%	3.07%	3.15%	3.11%	2.83%
	3	5.82%	6.91%	6.69%	6.14%	6.35%	6.08%	5.50%	6.28%	6.50%

Heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/40)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	5	12.08%	12.76%	12.12%	12.32%	11.96%	11.76%	11.62%	12.21%	11.82%
	10	24.19%	24.61%	23.16%	22.71%	22.76%	22.12%	21.90%	22.09%	21.64%
	30	43.83%	39.91%	36.83%	35.59%	34.78%	33.78%	33.91%	34.76%	33.46%
	60	57.75%	49.26%	45.88%	42.73%	40.97%	38.98%	38.47%	38.93%	38.21%
	Average	**20.97%**	**19.51%**	**18.19%**	**17.43%**	**17.12%**	**16.54%**	**16.37%**	**16.77%**	**16.35%**

Table A.12 Percentage optimality gaps of the heuristics for the case of a turnover-based storage policy with a 20/60 demand skewness in leaf warehouse.

Heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/60)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
LM	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	11.87%	10.44%	11.36%	10.42%	12.87%	11.89%	12.98%	13.33%	15.46%
	3	22.61%	18.08%	18.62%	19.26%	19.51%	20.69%	20.42%	22.50%	23.76%

Heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/60)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	5	41.64%	32.58%	29.21%	30.88%	30.66%	30.90%	31.11%	33.87%	33.96%
	10	66.95%	48.51%	41.71%	39.99%	39.18%	37.85%	38.01%	37.79%	37.25%
	30	73.79%	59.25%	45.88%	38.68%	35.44%	32.70%	30.44%	28.99%	27.07%
	60	80.41%	64.05%	52.02%	42.79%	37.22%	32.87%	29.75%	27.48%	25.01%
	Average	**42.47%**	**33.27%**	**28.40%**	**26.00%**	**24.98%**	**23.84%**	**23.24%**	**23.42%**	**23.21%**
LL	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	10.11%	9.88%	11.02%	10.84%	13.29%	11.67%	13.17%	13.32%	15.35%
	3	17.84%	16.44%	17.69%	19.71%	19.77%	20.34%	20.51%	22.34%	23.71%
	5	30.05%	27.04%	26.11%	28.61%	29.80%	29.17%	30.36%	33.02%	33.62%
	10	42.38%	38.30%	35.55%	36.23%	36.59%	35.18%	36.24%	36.30%	35.92%
	30	52.89%	46.09%	37.54%	32.66%	30.36%	28.46%	27.02%	25.61%	24.57%
	60	58.52%	47.28%	38.80%	32.50%	28.32%	25.16%	23.14%	21.30%	19.64%

Heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/60)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	Average	**30.25%**	**26.43%**	**23.82%**	**22.94%**	**22.59%**	**21.43%**	**21.49%**	**21.70%**	**21.83%**
LS	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	0.37%	1.88%	3.29%	4.91%	5.96%	6.94%	7.78%	7.98%	8.51%
	3	0.78%	3.86%	6.64%	8.68%	10.13%	12.35%	12.88%	14.70%	14.37%
	5	1.58%	6.18%	11.13%	14.18%	16.87%	18.34%	20.66%	22.65%	23.35%
	10	3.51%	9.85%	15.45%	20.24%	24.10%	27.29%	30.62%	33.21%	34.57%
	30	9.17%	10.65%	15.63%	20.40%	25.68%	29.78%	34.31%	37.97%	41.35%
	60	15.08%	8.15%	11.56%	14.41%	18.55%	22.07%	26.49%	29.89%	32.98%
	Average	**4.36%**	**5.80%**	**9.10%**	**11.83%**	**14.47%**	**16.68%**	**18.96%**	**20.92%**	**22.16%**
LR	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	1.81%	1.69%	1.73%	1.37%	1.40%	1.41%	1.17%	1.53%	1.58%
	3	3.23%	3.63%	3.30%	2.81%	2.99%	2.80%	2.88%	2.88%	2.56%

Heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/60)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	5	7.10%	6.84%	6.48%	6.25%	6.34%	5.97%	5.81%	6.25%	5.89%
	10	15.97%	14.41%	12.63%	11.70%	11.79%	11.53%	11.17%	11.64%	10.91%
	30	38.64%	29.51%	25.50%	23.28%	22.11%	21.49%	20.79%	20.76%	20.16%
	60	57.72%	41.68%	35.58%	31.64%	29.20%	27.65%	26.48%	26.20%	25.16%
	Average	**17.78%**	**13.97%**	**12.17%**	**11.01%**	**10.55%**	**10.12%**	**9.76%**	**9.89%**	**9.47%**

Table A.13 Percentage optimality gaps of the heuristics for the case of a turnover-based storage policy with a 20/80 demand skewness in leaf warehouse.

Heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/80)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
LM	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	5.91%	6.73%	8.01%	6.98%	11.45%	9.69%	11.69%	15.40%	14.38%
	3	13.59%	11.84%	13.27%	16.64%	18.56%	18.49%	21.16%	22.11%	22.50%

Heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/80)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	5	29.71%	21.36%	22.66%	28.34%	28.27%	31.45%	33.26%	32.01%	37.69%
	10	67.38%	43.11%	38.61%	39.32%	41.30%	41.55%	43.09%	43.52%	45.65%
	30	73.65%	60.48%	47.24%	42.81%	39.62%	36.99%	35.74%	34.77%	33.66%
	60	80.38%	63.33%	50.37%	41.55%	36.96%	32.98%	30.32%	28.65%	26.86%
	Average	**38.66%**	**29.55%**	**25.74%**	**25.09%**	**25.17%**	**24.45%**	**25.04%**	**25.21%**	**25.82%**
LL	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	4.76%	5.88%	7.89%	7.08%	12.10%	10.12%	11.83%	15.79%	14.38%
	3	8.91%	10.17%	12.76%	17.01%	18.49%	18.66%	20.88%	22.65%	22.37%
	5	16.62%	16.41%	20.25%	27.08%	27.52%	30.96%	32.48%	31.58%	37.53%
	10	31.20%	31.33%	32.63%	36.54%	38.58%	39.68%	41.68%	41.42%	44.19%
	30	52.54%	46.95%	39.84%	37.78%	35.53%	33.84%	33.19%	32.29%	31.46%
	60	58.53%	46.63%	38.10%	33.23%	30.09%	27.36%	25.79%	24.44%	23.02%

Heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/80)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	Average	**24.65%**	**22.48%**	**21.64%**	**22.67%**	**23.19%**	**22.95%**	**23.69%**	**24.02%**	**24.71%**
LS	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	0.44%	3.27%	5.42%	9.44%	12.40%	13.33%	15.61%	13.03%	16.00%
	3	0.77%	6.29%	11.01%	15.08%	18.97%	22.72%	25.19%	25.49%	28.29%
	5	1.11%	8.25%	16.91%	22.92%	28.50%	32.81%	36.04%	38.82%	40.28%
	10	2.27%	10.59%	20.77%	28.81%	34.97%	41.18%	44.68%	48.79%	51.46%
	30	8.02%	10.19%	17.58%	24.86%	31.34%	37.82%	42.88%	48.95%	53.28%
	60	15.10%	8.14%	12.54%	17.71%	23.09%	29.05%	33.58%	38.90%	43.59%
	Average	**3.96%**	**6.68%**	**12.04%**	**16.97%**	**21.32%**	**25.27%**	**28.28%**	**30.57%**	**33.27%**
LR	1	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	2	0.55%	0.62%	0.52%	0.40%	0.32%	0.41%	0.42%	0.41%	0.44%
	3	1.59%	1.48%	1.04%	0.86%	1.06%	0.92%	1.08%	1.07%	0.83%

Heuristics	Pick-list sizes	Percentage optimality gaps of the heuristics in different warehouse sizes using turnover-based storage policy (20/80)								
		WZ1	**WZ2**	**WZ3**	**WZ4**	**WZ5**	**WZ6**	**WZ7**	**WZ8**	**WZ9**
	5	3.33%	2.67%	2.21%	2.23%	2.13%	2.00%	2.28%	2.43%	1.80%
	10	9.82%	6.85%	5.50%	5.00%	4.92%	4.32%	4.73%	4.79%	4.27%
	30	35.64%	21.32%	15.64%	13.47%	11.96%	10.91%	10.64%	10.48%	10.13%
	60	57.73%	36.66%	26.91%	21.95%	19.62%	17.58%	16.37%	15.80%	14.74%
	Average	**15.52%**	**9.94%**	**7.40%**	**6.27%**	**5.71%**	**5.16%**	**5.07%**	**5.00%**	**4.60%**

References

Applegate, D., Bixby, R., Chvátal, V., & Cook, W. (2008). Concorde TSP solver. URL <http://www.tsp.gatech.edu/concorde/>.

Ardjmand, E., Shakeri, H., Singh, M., & Bajgiran, O. S. (2018). Minimizing order picking makespan with multiple pickers in a wave picking warehouse. *International Journal of Production Economics*, *206*, 169-183.

Ardjmand, E., Bajgiran, O. S., & Youssef, E. (2019). Using list-based simulated annealing and genetic algorithm for order batching and picker routing in put wall based picking systems. *Applied Soft Computing Journal*, *75*, 106-119.

Armstrong, R. D., Cook, W. D., & Saipe, A. L. (1979). Optimal batching in a semi-automated order picking system. *Journal of the Operational Research Society*, *30*(8), 711-720.

Baker, P., & Canessa, M. (2009). Warehouse design: A structured approach. *European journal of operational research*, *193*(2), 425-436.

Barachet, L. L. (1957). Letter to the Editor-Graphic Solution of the Traveling-Salesman Problem. *Operations Research*, *5*(6), 841-845.

Bartholdi, J. J., & Ratliff, H. D. (1978). Unnetworks, with applications to idle time scheduling. *Management Science*, *24*(8), 850-858.

Bender, P. S. (1981). Mathematical modeling of the 20/80 rule: theory and practice. Journal of Business Logistics, 2(2), 139-157.

Bódis, T., & Botzheim, J. (2018). Bacterial Memetic Algorithms for Order Picking Routing Problem with Loading Constraints. *Expert Systems with Applications*, *105*, 196-220.

Boysen, N., Briskorn, D., & Emde, S. (2017). Parts-to-picker based order processing in a rack-moving mobile robots environment. *European Journal of Operational Research*, *262*(2), 550-562.

Boysen, N., de Koster, R., & Weidinger, F. (2019). Warehousing in the e-commerce era: A survey. *European Journal of Operational Research*, *277*(2), 396-411.

Bukchin, Y., Khmelnitsky, E., & Yakuel, P. (2012). Optimizing a dynamic order-picking process. *European Journal of Operational Research*, *219*(2), 335-346.

Cambazard, H., & Catusse, N. (2018). Fixed-parameter algorithms for rectilinear Steiner tree and rectilinear traveling salesman problem in the plane. *European Journal of Operational Research*, *270*(2), 419-429.

Caron, F., Marchet, G., & Perego, A. (2000). Optimal layout in low-level picker-to-part systems. *International Journal of Production Research*, *38*(1), 101-117.

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

Çelik, M., & Süral, H. (2016). Order picking in a parallel-aisle warehouse with turn penalties. *International Journal of Production Research*, *54*(14), 4340-4355.

Çelik, M., & Süral, H. (2019). Order picking in parallel-aisle warehouses with multiple blocks: complexity and a graph theory-based heuristic. *International Journal of Production Research*, *57*(3), 888-906.

Chabot, T., Lahyani, R., Coelho, L. C., & Renaud, J. (2017). Order picking problems under weight, fragility and category constraints. *International Journal of Production Research*, *55*(21), 6361-6379.

Charkhgard, H., & Savelsbergh, M. (2015). Efficient algorithms for travelling salesman problems arising in warehouse order picking. *The ANZIAM Journal*, *57*(2), 166-174.

Chen, F., Wang, H., Qi, C., & Xie, Y. (2013). An ant colony optimization routing algorithm for two order pickers with congestion consideration. *Computers & Industrial Engineering*, *66*(1), 77-85.

Chen, F., Wang, H., Xie, Y., & Qi, C. (2016). An ACO-based online routing method for multiple order pickers with congestion consideration in warehouse. *Journal of Intelligent Manufacturing*, *27*(2), 389-408.

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

Chen, F., Xu, G., & Wei, Y. (2019a). Heuristic routing methods in multiple-block warehouses with ultra-narrow aisles and access restriction. *International Journal of Production Research*, *57*(1), 228-249.

Chen, F., Xu, G., & Wei, Y. (2019b). An integrated metaheuristic routing method for multiple-block warehouses with ultranarrow aisles and access restriction. Complexity, https://doi.org/10.1155/2019/1280285.

Chen, T. L., Cheng, C. Y., Chen, Y. Y., & Chan, L. K. (2015). An efficient hybrid algorithm for integrated order batching, sequencing and routing problem. *International Journal of Production Economics*, *159*, 158-167.

Cheng, C. Y., Chen, Y. Y., Chen, T. L., & Yoo, J. J. W. (2015). Using a hybrid approach based on the particle swarm optimization and ant colony optimization to solve a joint order batching and picker routing problem. *International Journal of Production Economics*, *170*, 805-814.

Chia Jane, C. (2000). Storage location assignment in a distribution center. *International Journal of Physical Distribution & Logistics Management*, *30*(1), 55-71.

Clarke, G., & Wright, J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research*, *12*(4), 568-581.

Cooper, H. (2010). *Research synthesis and meta-analysis: A step-by-step approach* (Vol. 2). Sage publications.

Cormier, G., & Gunn, E. A. (1992). A review of warehouse models. *European Journal of Operational Research*, *58*(1), 3-13.

Cortés, P., Gómez-Montoya, R. A., Muñuzuri, J., & Correa-Espinal, A. (2017). A tabu search approach to solving the picking routing problem for large-and medium-size distribution centres considering the availability of inventory and K heterogeneous material handling equipment. *Applied Soft Computing*, *53*, 61-73.

Croes, G. A. (1958). A method for solving traveling-salesman problems. *Operations research*, *6*(6), 791-812.

Daniels, R. L., Rummel, J. L., & Schantz, R. (1998). A model for warehouse order picking. *European Journal of Operational Research*, *105*(1), 1-17.

Davarzani, H., & Norrman, A. (2015). Toward a relevant agenda for warehousing research: literature review and practitioners' input. *Logistics Research*, *8*(1), 1.

De Koster, M. B. M., Van der Poort, E. S., & Wolters, M. (1999). Efficient orderbatching methods in warehouses. *International Journal of Production Research*, *37*(7), 1479-1504.

De Koster, R. B., Le-Duc, T., & Zaerpour, N. (2012). Determining the number of zones in a pick-and-sort order picking system. *International Journal of Production Research*, *50*(3), 757-771.

De Koster, R., & Van der Poort, E. (1998). Routing orderpickers in a warehouse: a comparison between optimal and heuristic solutions. *IIE Transactions*, *30*(5), 469-480.

De Koster, R., Le-Duc, T., & Roodbergen, K. J. (2007). Design and control of warehouse order picking: A literature review. *European Journal of Operational Research*, *182*(2), 481-501.

De Santis, R., Montanari, R., Vignali, G., & Bottani, E. (2018). An adapted ant colony optimization algorithm for the minimization of the travel distance of pickers in manual warehouses. *European Journal of Operational Research*, *267*(1), 120-137.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische mathematik*, *1*(1), 269-271.

Dukic, G., & Opetuk, T. (2012). Warehouse layouts. In Warehousing in the Global Supply Chain (pp. 55-69). Springer, London.

Elbert, R. M., Franzke, T., Glock, C. H., & Grosse, E. H. (2017). The effects of human behavior on the efficiency of routing policies in order picking: the case of route deviations. *Computers & Industrial Engineering*, *111*, 537-551.

Elsayed, E. A., & Stern, R. G. (1983). Computerized algorithms for order processing in automated warehousing systems. *International Journal of Production Research*, *21*(4), 579-586.

Fahimnia, B., Sarkis, J., & Davarzani, H. (2015). Green supply chain management: A review and bibliometric analysis. *International Journal of Production Economics*, 162, 101-114.

Fazlollahtabar, H., Saidi-Mehrabad, M., & Balakrishnan, J. (2015). Mathematical optimization for earliness/tardiness minimization in a multiple automated guided vehicle manufacturing system via integrated heuristic algorithms. *Robotics and Autonomous Systems*, *72*, 131-138.

Floyd, R. W. (1962). Algorithm 97: shortest path. *Communications of the ACM*, *5*(6), 345.

Franzke, T., Grosse, E. H., Glock, C. H., & Elbert, R. (2017). An investigation of the effects of storage assignment and picker routing on the occurrence of picker blocking in manual picker-to-parts warehouses. *The International Journal of Logistics Management*, 28(3), 841-863.

Frazelle, E. (2002). World-Class Warehousing and Material Handling. McGraw-Hill. *New York*, *NY*.

Fumi, A., Scarabotti, L., & Schiraldi, M. M. (2013). Minimizing warehouse space with a dedicated storage policy. *International Journal of Engineering Business Management*, *5*(Godište 2013), 5-21.

Gademann, A. N. (1999). Optimal routing in an automated storage/retrieval system with dedicated storage. *IIE Transactions*, *31*(5), 407-415.

Gademann, A.J.R.M., Van Den Berg, J. P., & Van Der Hoff, H. H. (2001). An order batching algorithm for wave picking in a parallel-aisle warehouse. *IIE Transactions*, *33*(5), 385-398.

Gademann, N., & van de Velde, S. (2005). Order batching to minimize total travel time in a parallel-aisle warehouse. *IIE Transactions*, *37*(1), 63-75.

Gerking, H. (2009). Kommissionierstrategien: Schleife, Stichgang, Walking the U. In: Pulverich M, Schietinger, J (eds.), Handbuch Kommissionierung. Vogel, München, pp. 148–155.

Gibson, D. R., & Sharp, G. P. (1992). Order batching procedures. *European Journal of Operational Research*, *58*(1), 57-67.

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

Glock, C. H., Grosse, E. H., Abedinnia, H., & Emde, S. (2019). An integrated model to improve ergonomic and economic performance in order picking by rotating pallets. *European Journal of Operational Research*, *273*(2), 516-534.

Glock, C. H., Grosse, E. H., Elbert, R. M., & Franzke, T. (2017). Maverick picking: the impact of modifications in work schedules on manual order picking processes. *International Journal of Production Research*, *55*(21), 6344-6360.

Goetschalckx, M., & Ratliff, H.D. (1988a). Order picking in an aisle. *IIE Transactions*, *20*(1), 53-62.

Goetschalckx, M., & Ratliff, H. D. (1988b). An efficient algorithm to cluster order picking items in a wide aisle. *Engineering Costs and Production Economics*, *13*(4), 263-271.

Gong, Y., & De Koster, R. (2008). A polling-based dynamic order picking system for online retailers. *IIE Transactions*, *40*(11), 1070-1082.

Grosse, E. H., Glock, C. H., & Ballester-Ripoll, R. (2014). A simulated annealing approach for the joint order batching and order picker routing problem with weight restrictions. *International Journal of Operations and Quantitative Management*, *20*(2), 65-83.

Grosse, E. H., Glock, C. H., & Neumann, W. P. (2017). Human factors in order picking: a content analysis of the literature. *International Journal of Production Research*, *55*(5), 1260-1276.

Grosse, E. H., Glock, C. H., Jaber, M. Y., & Neumann, W. P. (2015). Incorporating human factors in order picking planning models: framework and research opportunities. *International Journal of Production Research*, 53(3), 695-717.

Gu, J., Goetschalckx, M., & McGinnis, L. F. (2007). Research on warehouse operation: A comprehensive review. *European Journal of Operational Research*, *177*(1), 1-21.

Gue, K. R., & Meller, R. D. (2009). Aisle configurations for unit-load warehouses. *IIE Transactions*, *41*(3), 171-182.

Gue, K. R., Ivanović, G., & Meller, R. D. (2012). A unit-load warehouse with multiple pickup and deposit points and non-traditional aisles. *Transportation Research Part E: Logistics and Transportation Review*, *48*(4), 795-806.

Hall, R. W. (1993). Distance approximations for routing manual pickers in a warehouse. *IIE Transactions*, *25*(4), 76-87.

Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, *4*(2), 100-107.

Helsgaun, K. (2000). An effective implementation of the Lin–Kernighan traveling salesman heuristic. *European Journal of Operational Research*, *126*(1), 106-130.

Henn, S. (2012). Algorithms for on-line order batching in an order picking warehouse. *Computers & Operations Research*, *39*(11), 2549-2563.

Henn, S., & Schmid, V. (2013). Metaheuristics for order batching and sequencing in manual order picking systems. *Computers & Industrial Engineering*, *66*(2), 338-351.

Henn, S., Koch, S., Doerner, K. F., Strauss, C., & Wäscher, G. (2010). Metaheuristics for the order batching problem in manual order picking systems. *Business Research*, *3*(1), 82-105.

Henn, S., Koch, S., Gerking, H., & Wäscher, G. (2013). A U-shaped layout for manual order-picking systems. *Logistics Research*, *6*(4), 245-261.

Ho, Y. C., & Tseng, Y. Y. (2006). A study on order-batching methods of order-picking in a distribution centre with two cross-aisles. *International Journal of Production Research*, *44*(17), 3391-3417.

Hochrein, S., & Glock, C. H. (2012). Systematic literature reviews in purchasing and supply management research: a tertiary study. *International Journal of Integrated Supply Management*, *7*(4), 215-245.

Hsieh, L. F., & Tsai, L. (2006). The optimum design of a warehouse system on order picking efficiency. *The International Journal of Advanced Manufacturing Technology*, *28*(5-6), 626-637.

Jane, C. C., & Laih, Y. W. (2005). A clustering algorithm for item assignment in a synchronized zone order picking system. *European Journal of Operational Research*, *166*(2), 489-496.

Jang, H. Y., & Sun, J. U. (2012). A Graph Optimization Algorithm for Warehouses with Middle Cross Aisles. *Applied Mechanics and Materials*, *145*, 354-358.

Jarvis, J. M., & McDowell, E. D. (1991). Optimal product layout in an order picking warehouse. *IIE Transactions*, *23*(1), 93-102.

Jünger, M., & Naddef, D. (Eds.). (2001). *Computational combinatorial optimization: optimal or provably near-optimal solutions* (Vol. 2241). Springer Science & Business Media.

Kulak, O., Sahin, Y., & Taner, M. E. (2012). Joint order batching and picker routing in single and multiple-cross-aisle warehouses using cluster-based tabu search algorithms. *Flexible Services and Manufacturing Journal*, *24*(1), 52-80.

Kunder, R., & Gudehus, T. (1975). Mittlere wegzeiten beim eindimensionalen kommissionieren. *Zeitschrift für Operations Research*, *19*(2), B53-B72.

Laporte, G. (1986). Generalized subtour elimination constraints and connectivity constraints. *Journal of the Operational Research Society*, *37*(5), 509-514.

Le-Duc, T., & de Koster, R. M. (2007). Travel time estimation and order batching in a 2-block warehouse. *European Journal of Operational Research*, *176*(1), 374-388.

Li, J., Huang, R., & Dai, J. B. (2017). Joint optimisation of order batching and picker routing in the online retailer's warehouse in China. *International Journal of Production Research*, *55*(2), 447-461.

Lin, C. C., Kang, J. R., Hou, C. C., & Cheng, C. Y. (2016). Joint order batching and picker Manhattan routing problem. *Computers & Industrial Engineering*, *95*, 164-174.

Lin, S. (1965). Computer solutions of the traveling salesman problem. *The Bell System Technical Journal*, *44*(10), 2245-2269.

Lin, S., & Kernighan, B. W. (1973). An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, *21*(2), 498-516.

Löffler, M., Boysen, N., & Schneider, M. (2018). Picker routing in AGV-assisted order picking systems, Working Paper, DPO-2018-01, Deutsche Post Chair-Optimization of Distribution Networks, RWTH Aachen University, 2018.

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

Makris, P. A., & Giakoumakis, I. G. (2003). k-Interchange heuristic as an optimization procedure for material handling applications. *Applied Mathematical Modelling*, *27*(5), 345-358.

Masae, M., Glock, C.H., & Grosse, E.H. (2019). Order picker routing in warehouses: A systematic literature review. *International Journal of Production Economics*, https://doi.org/10.1016/j.ijpe.2019.107564.

Masae, M., Glock, C.H., & Vichitkunakorn, P. (2019). Optimal order picker routing in the chevron warehouse. *IISE Transactions*, https://doi.org/10.1080/24725854.2019.1660833.

Masae, M., Glock, C.H., & Vichitkunakorn, P. (2019). Optimal picker routing in a conventional warehouse with two blocks and arbitrary starting and ending points of a tour. Working paper, Technischen Universität Darmstadt, Germany.

Masae, M., Glock, C.H., & Vichitkunakorn, P. (2019). A method for efficiently routing order pickers in the leaf warehouse. Working paper, Technischen Universität Darmstadt, Germany.

Matusiak, M., de Koster, R., & Saarinen, J. (2017). Utilizing individual picker skills to improve order batching in a warehouse. *European Journal of Operational Research*, *263*(3), 888-899.

Matusiak, M., de Koster, R., Kroon, L., & Saarinen, J. (2014). A fast simulated annealing method for batching precedence-constrained customer orders in a warehouse. *European Journal of Operational Research*, *236*(3), 968-977.

Menéndez, B., Pardo, E. G., Alonso-Ayuso, A., Molina, E., & Duarte, A. (2017). Variable neighborhood search strategies for the order batching problem. *Computers & Operations Research*, *78*, 500-512.

Mowrey, C. H., & Parikh, P. J. (2014). Mixed-width aisle configurations for order picking in distribution centers. *European Journal of Operational Research*, *232*(1), 87-97.

Nemhauser, G. L., & Wolsey, L. A. (1988). Integer and Combinatorial Optimization. Interscience Series in Discrete Mathematics and Optimization. *ed: John Wiley & Sons*.

Or, I., Traveling Salesman-Type Combinatorial Problems and Their Relation to the Logistics of Regional Blood Banking. Ph.D. Thesis, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, 1976.

Öztürkoğlu, Ö., & Hoser, D. (2019). A discrete cross aisle design model for order-picking warehouses. *European Journal of Operational Research*, *275*(2), 411-430.

Öztürkoğlu, Ö., Gue, K. R., & Meller, R. D. (2012). Optimal unit-load warehouse designs for single-command operations. *IIE Transactions*, *44*(6), 459-475.

Öztürkoğlu, Ö., Gue, K. R., & Meller, R. D. (2014). A constructive aisle design model for unit-load warehouses with multiple pickup and deposit points. *European Journal of Operational Research*, *236*(1), 382-394.

Pan, J. C. H., Shih, P. H., & Wu, M. H. (2015). Order batching in a pick-and-pass warehousing system with group genetic algorithm. *Omega*, *57*, 238-248.

Pansart, L., Catusse, N., & Cambazard, H. (2018). Exact algorithms for the order picking problem. *Computers & Operations Research*, *100*, 117-127.

Parikh, P. J., & Meller, R. D. (2010). A travel-time model for a person-onboard order picking system. *European Journal of Operational Research*, *200*(2), 385-394.

Parikh, P. J., & Meller, R. D. (2008). Selecting between batch and zone order picking strategies in a distribution center. *Transportation Research Part E: Logistics and Transportation Review*, *44*(5), 696-719.

Petersen, C. G. (1997). An evaluation of order picking routeing policies. *International Journal of Operations & Production Management*, *17*(11), 1098-1111.

Petersen, C. G. (1999). The impact of routing and storage policies on warehouse efficiency. *International Journal of Operations & Production Management*, *19*(10), 1053-1064.

Petersen, C. G., & Aase, G. (2004). A comparison of picking, storage, and routing policies in manual order picking. *International Journal of Production Economics*, *92*(1), 11-19.

Petersen, C. G., & Schmenner, R. W. (1999). An evaluation of routing and volume-based storage policies in an order picking operation. *Decision Sciences*, *30*(2), 481-501.

Petersen, C. G., Aase, G. R., & Heiser, D. R. (2004). Improving order-picking performance through the implementation of class-based storage. *International Journal of Physical Distribution & Logistics Management*, *34*(7), 534-544.

Pferschy, U., & Schauer, J. (2018). Order batching and routing in a non-standard warehouse. *Electronic Notes in Discrete Mathematics*, *69*, 125-132.

Picard, J. C., & Queyranne, M. (1978). The time-dependent traveling salesman problem and its application to the tardiness problem in one-machine scheduling. *Operations Research*, *26*(1), 86-110.

Pohl, L. M., Meller, R. D., & Gue, K. R. (2009). Optimizing fishbone aisles for dual-command operations in a warehouse. *Naval Research Logistics (NRL)*, *56*(5), 389-403.

Pohl, L. M., Meller, R. D., & Gue, K. R. (2011). Turnover-based storage in non-traditional unit-load warehouse designs. *IIE Transactions*, *43*(10), 703-720.

Potvin, J. Y., & Rousseau, J. M. (1993). A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research*, *66*(3), 331-340.

Ratliff, H. D., & Rosenthal, A. S. (1983). Order-picking in a rectangular warehouse: a solvable case of the traveling salesman problem. *Operations Research*, *31*(3), 507-521.

Richards, G. (2018). Warehouse management: a complete guide to improving efficiency and minimizing costs in the modern warehouse. Kogan Page Limited.

Roodbergen, K. J., & De Koster, R. (2001a). Routing order pickers in a warehouse with a middle aisle. *European Journal of Operational Research*, *133*(1), 32-43.

Roodbergen, K. J., & De Koster, R. (2001b). Routing methods for warehouses with multiple cross aisles. *International Journal of Production Research*, *39*(9), 1865-1883.

Roodbergen, K. J., & Vis, I. F. (2006). A model for warehouse layout. *IIE Transactions*, *38*(10), 799-811.

Roodbergen, K. J., Sharp, G. P., & Vis, I. F. (2008). Designing the layout structure of manual order picking areas in warehouses. *IIE Transactions*, *40*(11), 1032-1045.

Roodbergen, K. J., Vis, I. F., & Taylor Jr, G. D. (2015). Simultaneous determination of warehouse layout and control policies. *International Journal of Production Research*, *53*(11), 3306-3326.

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

Rouwenhorst, B., Reuter, B., Stockrahm, V., van Houtum, G. J., Mantel, R. J., & Zijm, W. H. (2000). Warehouse design and control: Framework and literature review. *European Journal of Operational Research*, *122*(3), 515-533.

Scholz, A., & Wäscher, G. (2017). Order Batching and Picker Routing in manual order picking systems: the benefits of integrated routing. *Central European Journal of Operations Research*, *25*(2), 491-520.

Scholz, A., Henn, S., Stuhlmann, M., & Wäscher, G. (2016). A new mathematical programming formulation for the Single-Picker Routing Problem. *European Journal of Operational Research*, *253*(1), 68-84.

Scholz, A., Schubert, D., & Wäscher, G. (2017). Order picking with multiple pickers and due dates–Simultaneous solution of Order Batching, Batch Assignment and Sequencing, and Picker Routing Problems. *European Journal of Operational Research*, *263*(2), 461-478.

Schrotenboer, A. H., Wruck, S., Roodbergen, K. J., Veenstra, M., & Dijkstra, A. S. (2017). Order picker routing with product returns and interaction delays. *International Journal of Production Research*, *55*(21), 6394-6406.

Segal, M. (1974). The operator-scheduling problem: A network-flow approach. *Operations Research*, *22*(4), 808-823.

Selvakumar, A. I., & Thanushkodi, K. (2007). A new particle swarm optimization solution to nonconvex economic dispatch problems. *IEEE Transactions on Power Systems*, *22*(1), 42-51.

Seuring, S., & Gold, S. (2012). Conducting content-analysis based literature reviews in supply chain management. *Supply Chain Management: An International Journal*, *17*(5), 544-555.

Shah, B., & Khanzode, V. (2017). A comprehensive review of warehouse operational issues. *International Journal of Logistics Systems and Management*, *26*(3), 346-378.

Shaw, P. (1997). A new local search algorithm providing high quality solutions to vehicle routing problems. *APES Group, Dept of Computer Science, University of Strathclyde, Glasgow, Scotland, UK*.

Shouman, M. A., Khater, M., & Boushaala, A (2007). Comparisons of order picking routing methods for warehouses with multiple cross aisles. *AEJ-Alexandria Engineering Journal*, *46*(3), 261-272.

Singh, K. N., & van Oudheusden, D. L. (1997). A branch and bound algorithm for the traveling purchaser problem. *European Journal of Operational Research*, *97*(3), 571-579.

Sörensen, K. (2015). Metaheuristics—the metaphor exposed. *International Transactions in Operational Research*, *22*(1), 3-18.

Stützle, T., & Hoos, H. H. (2000). MAX–MIN ant system. *Future Generation Computer Systems*, *16*(8), 889-914.

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

Theys, C., Bräysy, O., Dullaert, W., & Raa, B. (2010). Using a TSP heuristic for routing order pickers in warehouses. *European Journal of Operational Research*, *200*(3), 755-763.

Tompkins, J. A., White, J. A., Bozer, Y. A., & Tanchoco, J. M. A. (2010). *Facilities planning*. John Wiley & Sons.

Toth, P., and Vigo, D. (2014). “The Family of Vehicle Routing Problem.” In *Vehicle Routing: Problems*, edited by Toth, P. and Vigo. D., 1–23., Methods, and Applications Philadelphia: MOS-SIAM Series on Optimization.

Tsai, C. Y., Liou, J. J., & Huang, T. M. (2008). Using a multiple-GA method to solve the batch picking problem: considering travel distance and order due time. *International Journal of Production Research*, *46*(22), 6533-6555.

Van Den Berg, J. P. (1999). A literature survey on planning and control of warehousing systems. *IIE Transactions*, *31*(8), 751-762.

Van Gils, T., Caris, A., Ramaekers, K., Braekers, K., & de Koster, R. B. (2019). Designing efficient order picking systems: The effect of real-life features on the relationship among planning problems. *Transportation Research Part E: Logistics and Transportation Review*, *125*, 47-73.

Van Gils, T., Ramaekers, K., Caris, A., & de Koster, R. B. (2018). Designing Efficient Order Picking Systems by Combining Planning Problems: State-of-the-art Classification and Review. *European Journal of Operational Research*, *267*(1), 1-15.

Van Nieuwenhuyse, I., & de Koster, R. B. (2009). Evaluating order throughput time in 2-block warehouses with time window batching. *International Journal of Production Economics*, *121*(2), 654-664.

Vaughan, T. S., & Petersen, C. G. (1999). The effect of warehouse cross aisles on order picking efficiency. *International Journal of Production Research*, *37*(4), 881-897.

Venkitasubramony, R., & Adil, G. K. (2016). Analytical models for pick distances in fishbone warehouse based on exact distance contour. *International Journal of Production Research*, *54*(14), 4305-4326.

Warshall, S. (1962). A theorem on boolean matrices. *Journal of the ACM (JACM)*, *9*(1), 11-12.

Weidinger, F. (2018). Picker routing in rectangular mixed shelves warehouses. *Computers & Operations Research*, *95*, 139-150.

Weidinger, F., Boysen, N., & Schneider, M. (2019). Picker routing in the mixed-shelves warehouses of e-commerce retailers. *European Journal of Operational Research*, *274*(2), 501-515.

Won, J., & Olafsson, S. (2005). Joint order batching and order picking in warehouse operations. *International Journal of Production Research*, *43*(7), 1427-1442.

Xu, X., Liu, T., Li, K., & Dong, W. (2014). Evaluating order throughput time with variable time window batching. *International Journal of Production Research*, *52*(8), 2232-2242.

Yu, M., & De Koster, R. B. (2009). The impact of order batching and picking area zoning on order picking system performance. *European Journal of Operational Research*, *198*(2), 480-490.

Zaerpour, N., de Koster, R. B., & Yu, Y. (2013). Storage policies and optimal shape of a storage system. *International Journal of Production Research*, *51*(23-24), 6891-6899.

Zhang, Z., Che, O., Cheang, B., Lim, A., & Qin, H. (2013). A memetic algorithm for the multiperiod vehicle routing problem with profit. *European Journal of Operational Research*, *229*(3), 573-584.

Zhou, L., Li, Z., Shi, N., Liu, S., & Xiong, K. (2019). Performance Analysis of Three Intelligent Algorithms on Route Selection of Fishbone Layout. Sustainability, 11(4), 1148.

Žulj, I., Glock, C. H., Grosse, E. H., & Schneider, M. (2018). Picker routing and storage-assignment strategies for precedence-constrained order picking. *Computers & Industrial Engineering*, *123*, 338-347.