Designing Optimal Layouts for Block Stacking Warehouses

by

Shahab Derhami

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
August 5, 2017

Keywords: warehouse design, block stacking, facility layout, layout design, space utilization, cross-aisles

Approved by

Jeffrey Smith, Chair, Professor of Industrial and Systems Engineering
Kevin Gue, Co-chair, Professor of Industrial Engineering, University of Louisville
Alice Smith, Professor of Industrial and Systems Engineering
Daniel Silva, Assistant Professor of Industrial and Systems Engineering

Abstract

Storing pallets of Stock Keeping Units (SKUs) on top of one another on a warehouse floor is known as *block stacking*. Although this storage system can be inexpensively implemented in any open area, it is challenging in terms of space planning. Designing an optimal layout for this storage system involves determining the optimal numbers of aisles and cross-aisles, bay depths, cross-aisle types, and their locations in the layout. The storage space is wasted in this system by a combination of honeycombing and accessibility aisles. Honeycombing refers to unoccupied pallet positions in a partially occupied lane that are only available to the SKU that has occupied the first pallet position of the lane. The accessibility waste refers to the space devoted to aisles and cross-aisles because they are not used for pallet storage. There is a trade-off between honeycombing and accessibility waste with respect to lane depths. Shallow lanes generate less honeycombing waste but impose more aisles to the layout, whereas the opposite is true for deep lanes. Cross-aisles improve transportation costs within the warehouse, but their devoted space contributes to the wasted space. Hence, both space utilization and transportation costs must be considered to study cross-aisles. This dissertation explores the above trade-offs and relations from three different perspectives: (1) it proposes a closed-form solution model to obtain optimal lane depths for block stacking in diverse manufacturing and non-manufacturing environments; (2) it studies the layout design problem from space utilization perspective, and proposes an optimal model to find space-efficient layouts; and, (3) it investigates the effects of cross-aisles on transportation costs and proposes a multi-objective model to design layouts that optimize both space utilization and transportation costs. The proposed models along with the highlighted future extensions provide proper tools to the companies to design efficient warehouses and implement a comprehensive foundation for further research.

Acknowledgments

This dissertation is dedicated to my faithful parents, Mohammad and Fatemeh, whose support, encouragement, and constant love have sustained me throughout my life. The completion of my PhD would not be possible without their unwavering supports. I am very grateful to my loving wife, Bahareh, for her supports and encouragements during my PhD.

I would like to express my sincere gratitude to my adviser, Dr. Jeffrey Smith, whose persistent supports and immense knowledge enriched this dissertation. His patience and understanding contributed to my personal and professional life.

I would also like to express my deepest appreciation to my co-adviser Dr. Kevin Gue, whose excellent guidance and expertise greatly assisted me in writing this dissertation. His motivation and enthusiasm have always encouraged me to continue my graduate journey.

I am very grateful to Dr. Alice Smith for enriching insights on this dissertation. I have had the pleasure to work with her on other research papers and have learned a lot from her.

I would also like to thank Dr. Daniel Silva, who was always willing to help and provided me with insightful comments.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem definition and research questions

Storing pallets of Stock Keeping Units (SKUs) on the floor of a warehouse is called *block stacking*. In this storage system, pallets are stacked on top of one another to their maximum stackable heights, which depend on the condition and height of the pallets, load weights, safety limits, clear height of the warehouse, and so on. These inexpensive storage systems do not require any storage racks and can be implemented in any open area (see Figure 1.1). Hence, they are widely used in manufacturing systems and distribution centers.

Space planning is challenging in these warehouses. They are mainly operated under one of two storage policies: *dedicated* or *shared*. In the dedicated policy, lanes are dedicated to SKUs, and each SKU is allowed to be stored only in its assigned lanes, whereas in the shared policy empty lanes are available to all SKUs (see Figure 1.2). Thus, shared policy is more

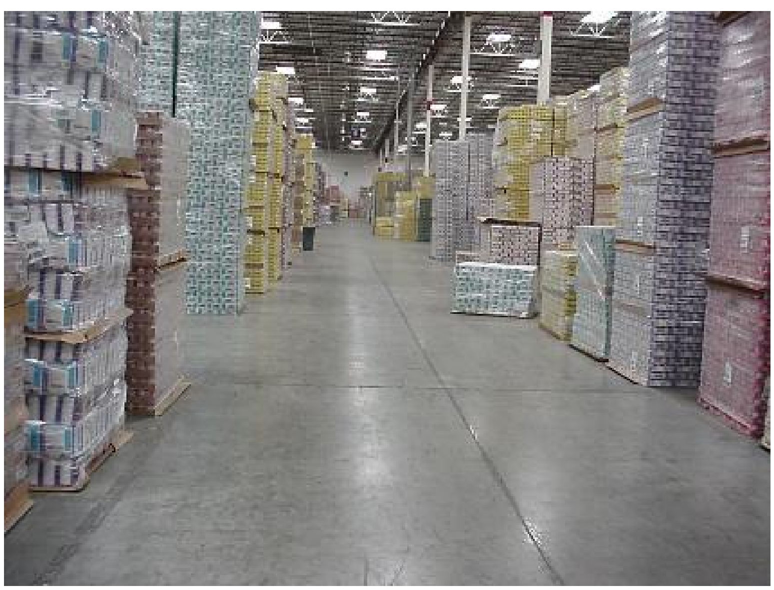

Figure 1.1: A block stacking warehouse.

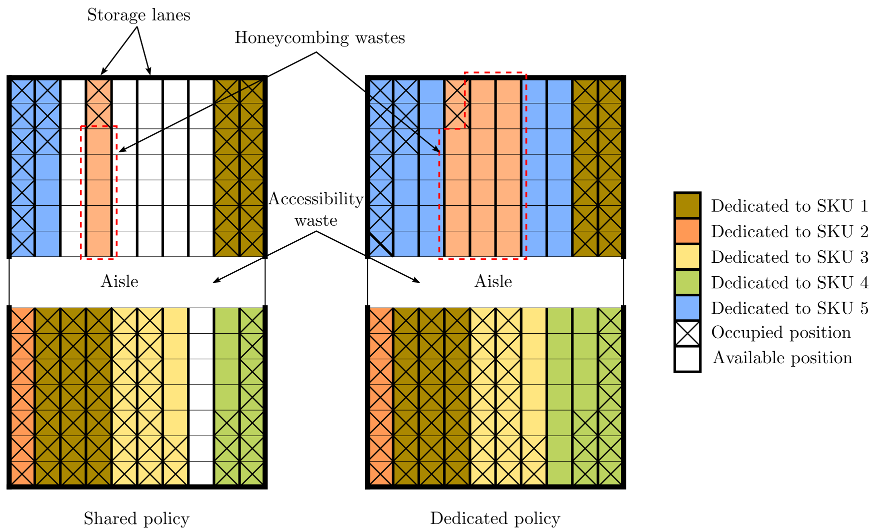

Figure 1.2: Dedicated policy vs. shared policy.

efficient in utilizing storage space but is less efficient in order picking as the result of variable lane assignments.

To avoid blockage or relocation of pallets in the shared policy, a lane is dedicated to a SKU once it occupies the first pallet position of the lane. This restriction wastes storage space because unoccupied pallet positions in a partially occupied lane are unavailable to other SKUs. This effect is called *honeycombing* and waste associated with it is incurred to the system until a lane becomes entirely occupied or emptied. In addition to honeycombing, aisles also contribute to the overall wasted space. They are required to access lanes, but their devoted space is not used directly for pallet storage. Figure 1.2 compares these two types of waste between the shared and dedicated policies.

There is a trade-off between honeycombing and accessibility wastes with respect to the lane depth. Deep lanes impose high honeycombing waste but produce fewer aisles in the layout, whereas the opposite is true for shallow lanes. As explained in [2], this trade-off must be optimized to minimize the total waste of storage space in a warehouse.

Cross-aisles reduce travel distances within a warehouse (see Figure 1.3) but, like aisles, they are not used for pallet storage and therefore, considered as a waste of storage space. It has been shown that there is a trade-off between space utilization and total travel distance

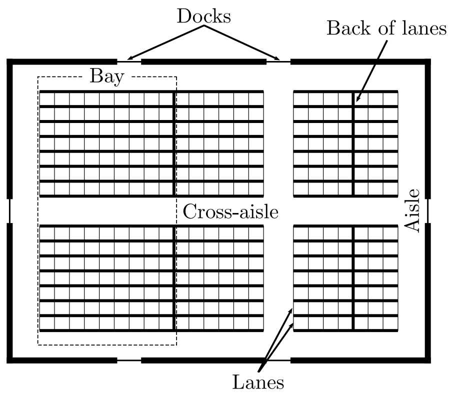

Figure 1.3: Components of a block stacking layout.

within the warehouse with respect to the layout bay depths [3]. The total travel distance decreases as bay depths decrease in the layout, whilst utilization of the storage space increases as bay depths increase.

The warehouse layout defines the shape, location, and size of the bays, lanes, aisles, and cross-aisles on the warehouse floor. The layout design problem is defined in this dissertation as determining the numbers and arrangement of bays, aisles, and cross-aisles, and the bay depths for the given warehouse area. To design an optimal layout, all aforementioned trade-offs must be considered. This dissertation aims to explore the layout design problem from both space utilization and transportation cost perspectives. It addresses the following major questions that have remained unanswered in the literature.

1. What is the lane depth that minimizes waste of storage space for block stacking in manufacturing environments?

2. How many aisles and bays should a layout have to maximize utilization of the storage space?

3. What are the optimal bay depths in a layout to efficiently utilize the storage space?

4. What is the most space-efficient SKU assignment policy?

5. How many cross-aisles and aisles should a layout have to minimize transportation costs and maximize space utilization?

1.2 Background

The research papers that investigated the warehouse layout problem mostly considered the conventional warehouses with storage racks [1, 6]. These studies mostly aimed to minimize transportation costs in order picking [7, 8, 12, 15]. The other objectives considered in designing a warehouse layout are operational costs [16], product allocation [11], warehouse throughput [13], and operating policies [14].

Few research papers studied designing a layout for block stacking. Kind [9] considered the trade-off between honeycombing and accessibility wastes to find the optimal lane depth that minimizes waste of storage space. However, he did not provide any derivations for his formula. Matson [10] developed another model to approximate the optimal lane depth under instantaneous replenishment (i.e., infinite storage rate) assumption. Her model is appropriate for warehouses that store products received from suppliers.

Goetschalckx and Ratliff [5] showed that if storage in multiple lane depths is allowed, the set of optimal lane depths follows a continuous triangular pattern. They developed a dynamic programming algorithm to select the set of optimal lane depths from a set of finite allowable lane depths to minimize the occupied floor space. Their approaches, especially the one that assumes unlimited multiple lane depths, are not practical for multiple SKUs.

This dissertation identifies three major gaps in the research focusing on block stacking warehouses.

1. All studies above assumed instantaneous lane replenishment [10, 2, 5]. In practice, this case only occurs in warehouses that store products received from suppliers. In such warehouses, a truck quickly unloads pallets and hence, it appears realistic to assume infinite arrival rate for incoming pallets. This is generally not true for warehouses located in manufacturing systems. In such systems, products are stored at finite rates, which are close to their production rates. The existing models cannot accurately address this prevalent case. The other restrictive assumption behind the existing optimal

lane depth model [10] is continuous demand. This limits using this model for seasonal products whose demands occur just at specific time periods, for instance, beginning of every month.

2. The research papers that studied block stacking warehouses focused exclusively on determining the lane depth that optimizes the trade-off between a block depth and width and do not provide any insights on designing the warehouse layout. To design a space-efficient layout, the number of aisles, bay depths, and SKU assignment policy must be determined. The optimal lane depth model [10, 5] cannot be used to find the optimal bay depth for a layout because it has a different waste function. It computes accessibility waste for the period that a lane is occupied. Therefore, it treats aisle space as a waste only when a lane is occupied and considers it as an available storage space otherwise. This is not true from the layout design perspective. In the layout design problem, the space dedicated to aisles is a permanent waste whether the adjoining lanes are occupied or not.

3. The transportation costs have not been studied adequately in designing the layout for block stacking warehouses. Most of the research projects in this field focus on improving utilization of space. The number of cross-aisles can be determined by considering the transportation costs. They reduce travel distance within the warehouse but wastes the storage space as well. It has been shown that there is a trade-off between space utilization and transportation costs with respect to bay depths [3]. Therefore, in order to find the optimal number of cross-aisles and aisles in the layout, both space utilization and transportation costs must be taken into account simultaneously.

This dissertation covers these gaps as follows. First, motivated by real-world problems, we extend the optimal lane depth model for manufacturing environments, where pallets are block stacked under various production conditions. For the first time, we evaluate the accuracy of the optimal lane depth model, which is built using deterministic assumptions, under stochastic conditions exist in real-world situations. We then explore the layout design problem and propose a model to design a space-efficient layout with multiple bay depths. Finally, we study the effects of the number of aisles and cross-aisles on transportation costs and space utilization and develop a model to design optimal layouts with respect to both of these objectives.

1.3 Contributions

This dissertation uses the following methodologies to address the current gaps in the literature. The second chapter explores the optimal lane depth model under various production and demand conditions. We relax the instantaneous resupply assumption made in the existing model [10] and develop closed-form solution models to obtain the optimal lane depth under finite production rate constraint. The proposed approach maximizes volume utilization in the warehouse rather than the floor area. We also relax the continuous demand assumption considered in [10]. That is, the storage (production) rate in our models can be finite and higher than the demand or less when the demand is discrete and intermittent. We develop a simulation model to evaluate performance of the proposed models under stochastic variations exist on the production and demand in real-world situations. The simulation results show that using infinite production rate model in a finite production rate system produces lane depths about twice as deep as they should be. However, the resulting waste of storage space is modest because the space utilization curve, as a function of lane depth, becomes quite flat as the lane depth grows [4].

The third chapter investigates the problem of designing a space-efficient layout for block stacking warehouses. In this chapter, we analyze waste of storage space from the layout design perspective and show that the waste function calculated by the optimal lane depth model underestimates the accessibility waste for the layout design purpose. We propose a new function to calculate waste of storage space with respect to bay depths and the number of aisles in the layout. We use a mixed integer programming model to optimize this function and to find the optimal number of aisles, bays, and bay depths in a layout. The model is NP-hard and highly symmetric. We propose various effective cuts to reduce the problem symmetry and to tighten the LP-relaxation lower bound. Our exhaustive experimental study that covers small to industrial-sized test problems shows that the proposed cuts effectively reduce the computational time of the proposed model. We found that accessibility waste outweighs honeycombing waste in their trade-off and an optimal model tends to choose deep lanes to reduce the number of aisles in the layout.

The fourth chapter studies the layout design problem from transportation cost perspective in addition to space utilization. We develop a simulation-based optimization model that finds the optimal number of aisles, cross-aisles, and cross-aisle types (unidirectional vs. bidirectional) that minimize transportation costs and maximize space utilization. The simulation model developed for this purpose is a discrete event-based model that simulates retrieval and replenishment operations in the warehouse. It benefits the proposed algorithm by incorporating the stochastic conditions that exist in real-world situations to the layout design problem. The simulation also accurately estimates the total travel distance for a multi-command operations environment (i.e. a warehouse where vehicles continue picking and delivering pallets without returning to their home or parking), where analytical models fail to provide accurate estimations. We develop a closed-form solution model to find the optimal number of aisles that maximizes space utilization under common bay depth constraint. Using this model along with numerical experiments, the model determines the simulation scenarios (layouts). Our exhaustive experimental study shows that the model is capable of exploring the Pareto front for industrial-sized layouts in a reasonable computational time.

1.4 Organization of the dissertation

The next chapters of this dissertation are organized as follows. First, the optimal lane depth models under finite production rate constraint are presented in chapter 2. Then, the proposed models to design a space-efficient layout is described in chapter 3. Next, the multi-objective model to optimize both space utilization and transportation costs in a layout is presented in chapter 4. Finally, the major findings and future research directions are highlighted in chapter 5. Chapters 2-4 start by introducing the research problems and describing the research motivations. Then, related research papers are reviewed, and the gaps in the literature and contributions are highlighted. Finally, the methodologies are presented, and the experimental studies and important findings are discussed.

Bibliography

[1] P. Baker and M. Canessa. Warehouse design: A structured approach. *European Journal of Operational Research*, 193(2):425–436, 2009.

[2] J. J. Bartholdi and S. T. Hackman. *Warehouse & Distribution Science: Release 0.89*. Supply Chain and Logistics Institute, 2008.

[3] S. Derhami, J. S. Smith, and K. R. Gue. A simulation model to evaluate the layout for block stacking warehouses. *Progress in Material Handling Research: 2016*, 2016.

[4] S. Derhami, J. S. Smith, and K. R. Gue. Optimising space utilisation in block stacking warehouses. *International Journal of Production Research*, pages 1–17, 2017. doi: 10.1080/00207543.2016.1154216.

[5] M. Goetschalckx and D. H. Ratliff. Optimal lane depths for single and multiple products in block stacking storage systems. *IIE Transactions*, 23(3):245–258, 1991.

[6] J. Gu, M. Goetschalckx, and L. F. McGinnis. Research on warehouse design and performance evaluation: A comprehensive review. *European Journal of Operational Research*, 203(3):539–549, 2010.

[7] K. R. Gue and R. D. Meller. Aisle configurations for unit-load warehouses. *IIE Transactions*, 41(3):171–182, 2009.

[8] K. R. Gue, G. Ivanović, and R. D. Meller. 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, 2012.

[9] D. Kind. Elements of space utilization. *Transportation and Distribution Management*, 15:29–34, 1975.

[10] J. O. Matson. *The analysis of selected unit load storage systems*. PhD thesis, Georgia Institute of Technology, Atlanta, GA, 1982.

[11] M. Moshref-Javadi and M. R. Lehto. Material handling improvement in warehouses by parts clustering. *International Journal of Production Research*, 54(14):4256–4271, 2016.

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

[13] J. A. Pazour and R. D. Meller. An analytical model for A-frame system design. *IIE Transactions*, 43(10):739–752, 2011.

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

[15] L. M. Thomas and R. D. Meller. Analytical models for warehouse configuration. *IIE Transactions*, 46(9):928–947, 2014.

[16] L. M. Thomas and R. D. Meller. Developing design guidelines for a case-picking warehouse. *International Journal of Production Economics*, 170, Part C:741–762, 2015.

Chapter 2

Optimizing space utilization in block stacking warehouses

2.1 Abstract

Block stacking storage is an inexpensive storage system widely used in manufacturing systems where pallets of stock keeping units (SKUs) are stored in a warehouse at the finite production rates. However, determining the optimal lane depth that maximizes space utilization under a finite production rate constraint has not been adequately addressed in the literature and is an open problem. In this research, we propose mathematical models to obtain the optimal lane depth for single and multiple SKUs where the pallet production rates are finite. A simulation model is used to evaluate performance of the proposed models under stochastic uncertainty in the major production parameters and the demand.

Keywords: block stacking; facility layout; optimal lane depth; warehouse design; space utilization

2.2 Introduction

Optimizing space utilization has been one of the main goals in designing and operating warehouses [29]. The U.S. Roadmap for Material Handling and Logistics recognizes low warehouse utilization as one of the main factors that propels companies, associations and governments to employ collaborative warehouses more in the next decade. It also predicts that requests for high speed delivery or same-day delivery forces companies to build their warehouses and distribution centers near major metropolitan area where real estate is very expensive and therefore efficient use of space becomes more important [10].

Various approaches from the design to the operational phase of a warehouse have been developed to better utilize storage space. Block stacking is an inexpensive and conventional storage system whose performance depends on the efficient use of space. It is a unit load storage system in which pallets of stock keeping units (SKUs) are stacked on top of one another in lanes on the warehouse floor. Pallets are stacked to the maximum stacking height which depends on the conditions and heights of the pallets, load weights, safety limits, clearance height of the warehouse, and so on. No racking or storage facility is required for this system and it can be employed in any warehouse with wide floor space. This makes it an inexpensive storage system to implement but challenging to manage in terms of space planning.

Two major operating policies that are widely used to manage storage spaces in this system are dedicated and shared storage policies. In the dedicated policy, lanes are dedicated to SKUs and only pallets of the assigned SKU are allowed to be stored in a lane. So, a lane may remain empty until it is replenished with its assigned SKU. On the other hand, lanes are not dedicated to any SKUs in the shared storage policy and they are available to all SKUs once they become empty. This policy utilizes space more efficiently than the former one though the order picking process is generally less efficient since the SKUs storage locations change over time and SKUs are assigned to the lanes based on their availability rather than convenience of their locations. However, shared storage is widely used in warehouses that deal with numerous SKUs and limited storage space.

The shared policy is operated with allowing or not allowing blockage. When the variety of SKUs is large but their inventory is small, assigning a lane to a single SKU is not justifiable and therefore different SKUs have to be stored in the same lane. In this case, blockage is inevitable and the goal is to arrange SKUs such that the relocation costs are minimized. An example of such a case is the storage space allocation problem in a marine container terminal [30, 23, 13, 4]. On the other hand, when the inventories of the SKUs are big enough to justify assigning a lane to a single SKU, no blockage policy is enforced. In this case to

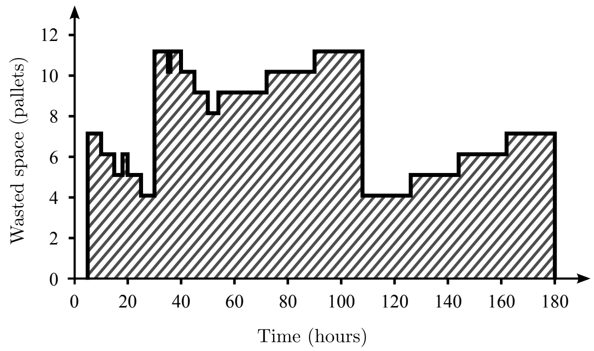

Figure 2.1: Waste of space in Example 2.1

avoid blockage and relocation, a lane is dedicated to a SKU once it occupies the first position of the lane. This case mainly occurs in the warehouses located in manufacturing systems or the distribution centers where plenty of pallets of different SKUs are block-stacked. However, this restriction wastes storage spaces in a lane when it is filled or depleted as there will be some unoccupied pallet positions in the lane that are just available to the pallets of the assigned SKU. This effect is termed honeycombing and waste associated with it is incurred to the system until a lane becomes entirely occupied or empty [12].

In addition to honeycombing, aisles also contribute to the overall wasted space. Aisles are required to have access to the lanes but their devoted spaces are not directly used for storage. Warehouse designers aim to minimize these two types of waste to enhance space utilization in the warehouse. The following example shows how waste of storage space is calculated in a lane.

Example 2.1. Consider a batch of 10 pallets of a SKU is stored in a lane of two pallets deep. Pallets are produced at the rate of (1/5) pallets per hour, stacked up to two pallets high and depleted at the rate of (1/18) pallets per hour. Assume that an aisle with width equivalent to two pallets is required to access the lane. Figure 2.1 shows waste of storage space generated in the inventory cycle time of this SKU. At time zero, waste is zero because an empty lane is available to all SKUs. At time five, the first pallet is stored in the lane and makes the three unoccupied pallet positions in the lane unavailable to the other SKUs.

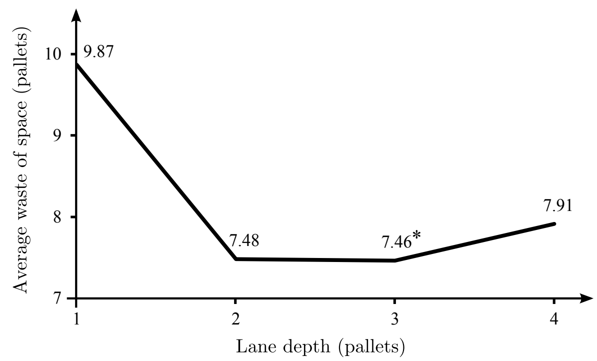

Figure 2.2: Optimal lane depth in Example 2.1.

This is the honeycombing waste. Moreover, four pallet positions are dedicated to the aisle to provide accessibility to the lane. So, seven pallet positions are wasted at this time. At times 10 and 15, the next two pallets are stored in the lane and the total number of wasted positions decreases to six and then five pallets, respectively. The first depletion event occurs at time 18 and increases the number of wasted positions to six. This procedure continues until all 10 pallets are produced, stored in the lane, and depleted. The area under the waste plot in Figure 2.1 is the pallets-time waste of storage space during the inventory cycle time, and dividing it by 175 hours gives the average waste of storage space in the inventory cycle time which is 7.48 pallets.

Storing a batch of pallets in deep lanes increases the honeycombing waste, but the space required for aisles decreases while the reverse is true for the shallow lanes. Hence, a trade-off between the lane depth and width must be considered to optimize space utilization. This trade-off is shown in Figure 2.2 by comparing the average waste of storage space generated by storing the SKU described in Example 2.1 in lanes with one to four pallets deep. In this case, the minimum waste achieved when the SKU is stored in the lanes with three pallets deep.

In this paper, we consider this trade-off from a mathematical point of view and develop models to compute an optimal lane depth that minimizes waste of storage space in the warehouse. Our models are different from those existing in the literature in two major respects. First, the instantaneous pallet storage assumption, which was made in all previous research is relaxed and models are built for finite production rates. Hence, they are more suitable for the warehouses located in manufacturing systems where pallets are stored at finite production rates. Second, our proposed models aim to maximize utilization of the volume instead of the floor space. A review of the previous research on block stacking is provided in the next section.

2.3 Related research

Various studies have investigated designing the layout of a warehouse [7]. Most of them considered designing the layout with respect to the construction and maintenance costs [1, 27, 22], material handling costs in order picking process [9, 24, 20, 21, 25], products allocation [11, 26] and few of them considered objectives pertinent to the space utilization [8]. Extensive reviews on different approaches used to design different storage systems are found in [7, 2, 6, 15, 28]. Studies that investigated space utilization in the block stacking storage systems are reviewed in the following.

To the best of our knowledge, Kind [14] was the first person who considered the trade-off between the lane depth and width in the block stacking storage and proposed a model to approximately find the optimal lane depth that minimizes waste of storage space. He proposed this approximation for a single SKU whose batch of pallets are instantaneously stored in a warehouse. Marsh [17] developed a simulation model to evaluate different storage and operating policies for block-stacking. Later, Matson [18] developed a more accurate version of Kind's approximation [14] for a single SKU and extended it to obtain the optimal common lane depth for multiple SKUs. Her models aim to maximize utilization of the floor space (area).

Goetschalckx and Donald Ratliff [5] showed that if a batch of pallets of a SKU is allowed to be stored in lanes with unlimited different depths, then the optimal lane depths follow a continuous triangular pattern. They developed a continuous and discrete approximations to obtain the optimal multiple lane depths considering limited and unlimited lane depths. They compared their proposed models with scenarios like equal lane depth models developed by Matson [18] and two extreme heuristics in which a batch of pallets is stored in the lanes whose depths are equal to one or to the batch size, respectively. They concluded space utilization is “relatively insensitive” to the lane depth and all heuristic methods except the two extreme cases provide comparatively equivalent results in terms of accuracy and computational complexity. However, their approaches, especially the one that assumes unlimited multiple lane depths, are not practical for multiple SKUs.

Larson et al. [16] proposed a heuristic approach to design the layout of a block stacking warehouse where the objectives are maximizing space utilization and minimizing transportation costs. Their class-based storage approach classifies SKUs based on the throughput to the required storage space ratio, ranks classes based on their average ratios, and constructs and dedicates the storage regions to the classes considering their ranks and required storage spaces. The algorithm considers honeycombing, fluctuations in the inventory level and the maximum stacking height to determine storage medium (i.e., racks or floor stacking) for a SKU, and assumes randomized storage policy among the classes (storage zones).

Bartholdi [3] developed Matson’s model to optimize volume utilization instead of the floor utilization. He suggested that maximizing the volume utilization is the better objective in the current modern warehouse because volume within a warehouse is worth as much as floor space in today’s modern warehouses.

All aforementioned studies assumed pallets of SKUs are instantaneously stored in a warehouse. In practice, this case occurs in a distribution center where trucks quickly unload pallets of SKUs, and hence it appears realistic to assume infinite arrival rate for incoming pallets. However, this assumption cannot be justified in the warehouses located in manufacturing systems where pallets of SKUs are stored in the warehouse at finite production rates. In such a warehouse, waste of storage space is generated either when a lane is filled or depleted. However, the traditional models are not capable of taking the first part into account and therefore do not correctly compute the optimal lane depth for such cases. We address this drawback in this paper by developing models to determine the optimal lane depth for both single and multiple SKUs under the finite production rate constraint.

2.4 Optimal lane depth

Minimizing the waste of storage space maximizes space utilization. So, one can obtain the optimal lane depth that maximizes the space utilization by deriving a mathematical expression to calculate waste of storage space and then finding the optimal lane depth that minimizes this waste expression. The waste of storage space is obtained by calculating three types of waste:

1. Waste of storage space caused by honeycombing, W_H.
2. Waste of storage space dedicated to the aisles, W_A.
3. Waste of unoccupied storage space on top of the occupied lanes, W_U.

W_U is incurred as the result of different stacking and pallet heights for different SKUs. This waste is not computed in the single SKU models as it does not affect the optimal lane depth in that cases. Figure 2.3 shows total waste of storage space and its components with respect to the optimal lane depth. The relation between the total waste of storage space and lane depth is analogous to the relation between the total cost and order quantity in the Economic Order Quantity (EOQ) model.

In general, the number of SKUs stored in a warehouse is too numerous to assign all SKUs to their optimal lane depths and assort all the lane depths in the warehouse. To overcome this issue, the optimal common lane depth is computed — one that minimizes total waste for multiple SKUs.

The models in this section are derived by assuming that a batch of Q pallets of a SKU is produced (or unloaded to a warehouse) at the deterministic production rate of P pallets per unit of time and block-stacked in the lanes of x pallets deep to the height of z pallets.

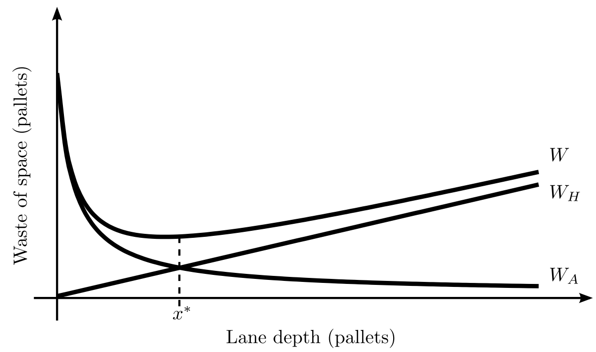

Figure 2.3: Total waste of storage space.

Pallets are depleted at the deterministic rate of λ pallets per unit of time and aisles with a pallets width are required to access the lanes. Table 2.1 describes the notation used in the following models. The following assumptions are made for all models in this paper:

1. Lanes are accessible from one side and as a result, they are depleted based on last-in-first-out (LIFO).

2. Partially occupied lanes are prioritized to be depleted first. This helps to utilize space more efficiently because unlike the fully occupied lanes which incur only accessibility wastes to the system, a partially occupied lane generates both honeycombing and accessibility waste. Thus, the longer it remains incomplete, the more storage space is wasted. Another advantage of this policy is that these lanes are depleted faster than the fully occupied lanes and consequently, their devoted spaces are released sooner.

3. Fully occupied lanes are depleted in an arbitrary order. This is because the order of depleting such lanes does not affect waste of storage space.

4. The production, if exists, and the depletion quantities are one pallet at a time and the storage, and depletion times are assumed to be zero.

5. No safety stock is kept in the warehouse. Production stops once all pallets of a batch are produced and it is restarted when the inventory of the SKU becomes zero.

Table 2.1: Table of notation.

P	production rate (in units of pallet/time)
λ	depletion rate (in units of pallet/time)
x	lane depth (in units of pallet)
x^*	optimal lane depth for single SKU (in units of pallet)
x_c^*	optimal common lane depth for multiple SKUs (in units of pallet)
z	stackable height (in units of pallet)
Q	production (arrival) batch quantity (in units of pallet)
H	maximum inventory level (approximation)
K	maximum number of lanes required for storage (approximation)
a	aisle width (in units of pallets)
h	height of a pallet of a SKU (in units of distance i.e., inch, cm)
e	clear height of the warehouse (in units of pallet)
n	number of SKUs
T	inventory cycle time
O_T	occupied space-time in the inventory cycle time
U	space utilization (single SKU)
U_c	space utilization for a common lane depth (multiple SKUs)
W_H	waste of storage space caused by honeycombing
W_A	waste of storage space dedicated to the aisles
W_U	waste of unoccupied storage space on top of the occupied lanes
W	average waste of storage space (single SKU)
W_c	average waste of storage space for a common lane depth (multiple SKUs)

2.4.1 $P = \infty$ demand is continuous

In this case, pallets of SKUs are instantaneously stored in a warehouse and depleted at a finite rate. So, the production rate is considered to be infinite. Figure 2.4 shows the inventory level of such a SKU over its inventory cycle time. Kind [14] proposed the following formula to estimate the optimal lane depth in this case:

$$x^* = \sqrt{\frac{Qa}{z}} - \frac{a}{2}. \tag{2.1}$$

However, he did not provide any derivation for his formula. Later, Matson [18] developed another approximation for the optimal lane depth. We derive the optimal lane depth for this case by providing a correction on the model proposed by Matson [18] and also considering volume utilization instead of the floor utilization as proposed by Bartholdi [3]. The correction herein is correcting the approach used to calculate total space dedicated to the aisles.

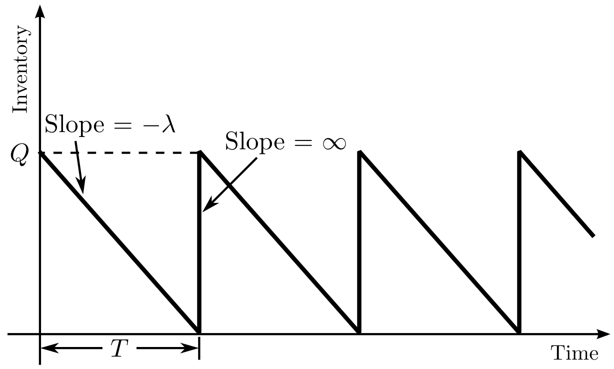

Figure 2.4: Changes in the inventory of a SKU stored instantaneously, $P = \infty$.

However, applying this correction does not change the original lane depth model proposed by Matson [18] for the single SKU, but the model for multiple SKUs changes as the result of optimizing volume utilization.

2.4.1.1 Optimal lane depth for a single SKU

The number of lanes required for storage is $\lceil Q/zx \rceil$, where $\lceil x \rceil$ is the smallest integer not less than x. Relaxing the integrality restriction, the approximate number of lanes would be

$$K \approx \frac{Q}{zx}. \tag{2.2}$$

Assume a fully occupied lane is being depleted at the rate of λ pallets per unit of time. Once the first pallet is depleted, the lane will have one unoccupied but unavailable position to the other SKUs. This waste of storage space remains in the lane for the time period of $(1/\lambda)$. Then the second pallet is depleted and two pallet positions are wasted for the same amount of time. This waste is rendered and accumulated until the last pallet is depleted at which $(zx - 1)$ pallet positions are unoccupied. Total pallets-time wasted in a lane as a result of honeycombing is

$$\left(\frac{1}{\lambda}\right)(1 + 2 + \cdots + (zx - 1)), \tag{2.3}$$

Multiplying (2.3) by approximate number of lanes gives

$$W_H \approx \left(\frac{1}{\lambda}\right)\left(\frac{Q(zx-1)}{2}\right). \tag{2.4}$$

W_A is calculated by computing total time that lanes are occupied and require accessibility. Herein, in accordance with assumption (2) in section (2.4), the last lane, which is most likely a partially occupied lane, is depleted first and the remaining lanes are depleted in an arbitrary order. The lane that is depleted at the last remains occupied until the whole batch is gone. So, this lane remains occupied for $(1/\lambda)(Q)$ time periods. The lane depleted before this lane becomes entirely empty $(1/\lambda)(zx)$ time periods before the latter one. Thus, it remains occupied for $(1/\lambda)(Q-zx)$ time periods. This process is applied to the remaining lanes, and the total time that all lanes are occupied is

$$\left(\frac{1}{\lambda}\right)\left(Q+(Q-zx)+(Q-2zx)+\cdots+(Q-Kzx)\right). \tag{2.5}$$

Each aisle is shared between two lanes located on both sides of it, so half of an aisle volume is dedicated to each lane. It is equal to $(az/2)$ pallets. This follows that

$$W_A \approx \left(\frac{1}{\lambda}\right)\left(Q(K+1)-\left(\frac{K(K+1)}{2}\right)zx\right)\left(\frac{az}{2}\right). \tag{2.6}$$

O_T is given by

$$\begin{aligned} O_T &\approx \left(\frac{1}{\lambda}\right)\left(Q+(Q-1)+\cdots+1\right), \\ &\approx \left(\frac{1}{\lambda}\right)\left(\frac{Q(Q+1)}{2}\right). \end{aligned} \tag{2.7}$$

Subsequently, space utilization is obtained by

$$U \approx \frac{O_T}{O_T + W_A + W_H}, \approx \frac{2x(Q+1)}{(2x+a)(Q+zx)}. \tag{2.8}$$

Finally, the average waste of storage space is obtained by summing (2.4) and (2.6) and dividing the result by T which is (Q/λ). That is,

$$W \approx \left(\frac{1}{4x}\right)\left(Qa - 2x + zx(2x+a)\right). \tag{2.9}$$

Proposition 2.1. *The optimal lane depth to block-stack a SKU whose batch of Q pallets is instantaneously stored in a warehouse is*

$$x^* \approx \sqrt{\frac{Qa}{2z}}. \tag{2.10}$$

Proof. The optimal lane depth is obtained by taking the derivative of (2.9) with respect to x, set it equal to zero and solve for x. □

From a practical point of view, the optimal lane depth should be an integer. To obtain an integer lane depth, the two nearest integers to x^* are obtained by rounding x^* up and down and then evaluating (2.9) at these values. This method can be used to obtain integer solutions for all further propositions.

2.4.1.2 Common optimal lane depth for multiple SKUs

W_U needs to be computed for the multiple SKUs model. If the clear height of a warehouse in units of pallets of a particular SKU is denoted by e, then $(e-z)x$ is the total pallet positions wasted on top of a lane (stack) for the period of time that the lane is occupied by that SKU. The total time that all lanes are occupied is given by (2.5). It follows that

$$W_U \approx \left(\frac{1}{\lambda}\right)\left(Q(K+1)-\left(\frac{K(K+1)}{2}\right)zx\right)(e-z)\,x. \tag{2.11}$$

Taking the clear height of the warehouse into account changes the space that a lane requires for accessibility to $(ae/2)$. So, expression (2.6) is updated accordingly.

Denote the height of a pallet of SKU i by h_i. Different SKUs may have different pallet heights and consequently be stackable to different heights. To take this into account, all waste expressions are scaled by multiplying to a factor of h_i. Total waste of storage space for each SKU is obtained by aggregating all three types of waste. Denote the least common multiple of the inventory cycle times of all SKUs by T_{LCM}. Since SKUs have different inventory cycle times, W_c is determined by calculating the total waste that all SKUs generate in T_{LCM} and then dividing the result by T_{LCM}. The number of inventory turns of SKU i in T_{LCM} is $T_{LCM}(\lambda_i/Q_i)$, and multiplying it by the total waste that SKU i generates in its inventory cycle time gives the total waste that SKU i generates in the T_{LCM}. Summing these wastes for all SKUs and dividing the result by the T_{LCM} results in the T_{LCM} terms to be canceled out from the expression. Therefore from a mathematical point of view, W_c is obtained by summing Ws for all SKUs.

$$W_c \approx \sum_{i=1}^{n}\left(\frac{h_i}{4z_i x}\right)\left(Q_i e_i(2x+a)+z_i x(2e_i x+ae_i-2Q_i-2)\right). \tag{2.12}$$

U_c is calculated similarly by computing the occupied and wasted space-time in T_{LCM}. That is,

$$\begin{aligned} U_c &\approx \frac{\sum_{i=1}^{n}\left(\frac{\lambda_i}{Q_i}\right)(O_T^i)}{\sum_{i=1}^{n}\left(\frac{\lambda_i}{Q_i}\right)(O_T^i+W_A^i+W_H^i)} \\ &\approx \frac{2x\sum_{i=1}^{n}h_i(Q_i+1)}{(2x+a)\sum_{i=1}^{n}e_i h_i\left(\frac{Q_i}{z_i}+x\right)}. \end{aligned} \tag{2.13}$$

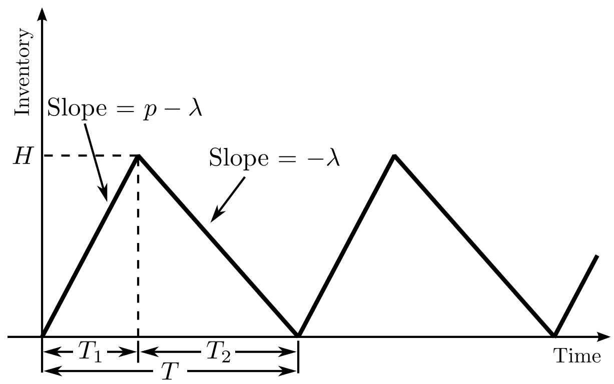

Figure 2.5: Changes in the inventory of a SKU stored at the rate P and depleted at the rate λ, where $P > \lambda$.

where W_H^i, W_A^i and O_T^i are obtained for SKU i by (2.4), (2.6) and (2.7), respectively. Note that T_{LCM} is canceled out in this expression too.

Proposition 2.2. *The optimal common lane depth to block-stack n SKUs whose batches of pallets are instantaneously stored in a warehouse is*

$$x_c^* \approx \sqrt{\frac{a \sum_{i=1}^{n} \left(\frac{e_i h_i}{z_i}\right) Q_i}{2 \sum_{i=1}^{n} e_i h_i}}. \tag{2.14}$$

Proof. Differentiating (2.12) with respect to x, setting it equal to zero and solving for x gives the result. □

2.4.2 $P > \lambda$ demand is continuous

This is a prevalent case in manufacturing systems where pallets of SKUs are produced at finite rates, stored in the warehouse and depleted at finite rates. Figure 2.5 shows changes in the inventory level of a SKU in this system. Period T_1 is the production phase in which Q pallets of the SKU are produced at the rate P and stored in the warehouse. Since the demand is constant, pallets are depleted at the rate λ in this period. To simplify calculations, we assume that lanes are filled at the rate $(P - \lambda)$. Production stops at the end of T_1 when the inventory reaches its maximum level. Then, the inventory starts decreasing in T_2 at the rate λ. Since the demand is constant, the inventory cycle time is (Q/λ), the maximum on-hand inventory, H, is $(Q - \lfloor Q\lambda/P \rfloor)$, and the maximum number of lanes required for storage is $(\lceil H/zx \rceil)$ where $\lfloor x \rfloor$ is the largest integer not greater than x. Relaxing the integrality restriction results in the following approximations:

$$H \approx \frac{Q\,(P-\lambda)}{P}, \tag{2.15}$$

$$K \approx \frac{Q\,(P-\lambda)}{Pzx}. \tag{2.16}$$

2.4.2.1 Optimal lane depth for a single SKU

W_H is generated in two phases, T_1 and T_2. First the former is calculated. Once a pallet is stored in an empty lane, $(zx-1)$ pallet positions are wasted in that lane for $1/(P-\lambda)$ time periods. This is the time that it takes until the next unoccupied position is filled. Then, $(zx-2)$ pallet positions are wasted for the same period of time. This waste is generated and accumulated until the last unoccupied pallet position in the lane is filled. So, the total honeycombing waste in a single lane in T_1 is

$$\left(\frac{1}{P-\lambda}\right)\left((zx-1)+(zx-2)+\cdots+(zx-(zx-1))\right). \tag{2.17}$$

Multiplying (2.17) by K gives

$$W_{H_{T_1}} \approx \left(\frac{(zx-1)}{2}\right)\left(\frac{Q}{P}\right). \tag{2.18}$$

$W_{H_{T_2}}$ is calculated similar to (2.3). Multiplying (2.3) by K results in

$$W_{H_{T_2}} \approx \left(\frac{zx-1}{2\lambda}\right)\left(\frac{Q\,(P-\lambda)}{P}\right). \tag{2.19}$$

W_A is calculated by approximating the total time that lanes are occupied in T_1 and T_2. Once the first pallet is stored in the first lane in T_1, this lane remains occupied until all $(H-1)$ positions are filled. Since the inventory increases at the rate $(P-\lambda)$, this lane remains occupied for $(H-1)/(P-\lambda)$ time periods. The second lane is used when the first one becomes fully occupied. It means $(H-zx)$ pallet positions are remained to be filled. Once the first pallet is stored in the second lane, this lane remains occupied until the remaining $(H-1-zx)$ pallet positions are filled. That is, it remains occupied for $(H-1-zx)/(P-\lambda)$ time periods. Thus, that the total time that all lanes are occupied in T_1 is

$$\left(\frac{1}{P-\lambda}\right)\left((H-1)+(H-1-zx)+(H-1-2zx)+\cdots+(H-Kzx)\right). \tag{2.20}$$

The total time that lanes are occupied in T_2 is calculated similar to (2.5) but herein H pallets are depleted, therefore Q is substituted with H in (2.5). Given that the dedicated aisle space to a lane is $(az/2)$,

$$W_A \approx \left(\left(\frac{1}{P-\lambda}+\frac{1}{\lambda}\right)\left(H\left(K+1\right)-zx\left(\frac{K\left(1+K\right)}{2}\right)\right)-\frac{K}{P-\lambda}\right)\left(\frac{az}{2}\right). \tag{2.21}$$

O_T is computed in T_1 and T_2 by

$$O_T \approx \left(\frac{1}{(P-\lambda)}\right)\left(1+2+\cdots+(H-1)\right)+\left(\frac{1}{\lambda}\right)\left(H+(H-1)+\cdots+1\right). \tag{2.22}$$

It follows that

$$U \approx \frac{2x(Q(P-\lambda)+P-2\lambda)}{(2x+a)(Q(P-\lambda)+Pzx-2\lambda)}. \tag{2.23}$$

The average waste of storage space is obtained by accumulating all types of waste and dividing the result by T, which is (Q/λ). That is,

$$W \approx \left(\frac{1}{4Px}\right)\left(2Px(zx-1)+aP(Q+zx)-a\lambda(Q+2)\right). \tag{2.24}$$

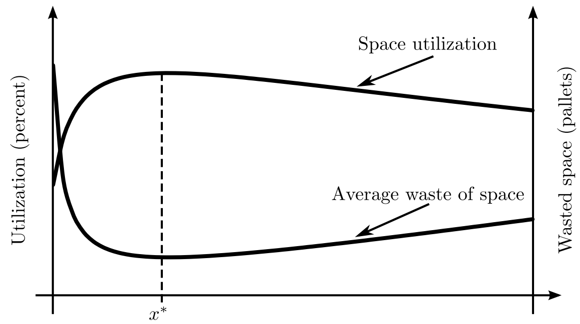

Figure 2.6: Space utilization vs. average waste of storage space.

Figure 2.6 compares space utilization and the average waste of storage space with respect to lane depth. At x^*, the space utilization and the average waste of storage space reach their maximum and minimum values, respectively.

Proposition 2.3. *The optimal lane depth to block-stack a SKU whose batch of pallets is produced at the rate P and depleted at the rate λ, where $P > \lambda$, is*

$$x^* \approx \sqrt{\frac{a\left(Q(P-\lambda)-2\lambda\right)}{2zP}}. \tag{2.25}$$

Proof. It is proved by taking the derivative of (2.24) with respect to x, setting it equal to zero and solving for x. □

2.4.2.2 Common optimal lane depth for multiple SKUs

W_U is calculated by multiplying the total time that lanes are occupied in T_1 and T_2 by the unoccupied volume on top of lanes which is $(e-z)x$ pallets. W_A is computed similar to (2.21), except that the aisle space that a lane requires for accessibility changes to $(ae/2)$. Scaling all waste expressions by multiplying them by h_i, summing them, and dividing the result by T results in W for SKU i. W_c is obtained by aggregating the Ws for all SKUs.

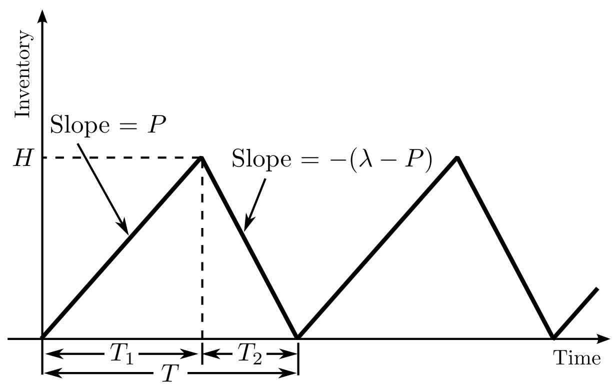

Figure 2.7: Changes in the inventory of a SKU stored at the rate P and depleted at the rate λ, where $\lambda > P$.

That is,

$$W_c \approx \sum_{i=1}^{n} \left(\frac{h_i}{4P_i z_i x} \right)$$
$$\left(P_i(Q_i e_i(2x+a) + z_i x(2e_i x - 2Q_i + ae_i - 2)) - \lambda(Q_i + 2)(2x(e_i - z_i) + ae_i)\right). \quad (2.26)$$

U_c is obtained as described in (2.13). That is,

$$U_c \approx \frac{2x \sum_{i=1}^{n} \left(\frac{h_i}{P_i}\right) (Q_i(P_i - \lambda_i) - 2\lambda_i + P_i)}{(2x+a) \sum_{i=1}^{n} e_i h_i (x + \left(\frac{1}{P_i z_i}\right) (Q_i(P_i - \lambda_i) - 2\lambda_i))}. \quad (2.27)$$

Proposition 2.4. *The optimal common lane depth to block-stack n SKUs whose batches of pallets are produced at the rate P and depleted at the rate λ, where $P > \lambda$, is*

$$x_c^* \approx \sqrt{\frac{a \sum_{i=1}^{n} \left(\frac{e_i h_i}{z_i P_i}\right) (Q_i(P_i - \lambda_i) - 2\lambda_i)}{2 \sum_{i=1}^{n} e_i h_i}}. \quad (2.28)$$

Proof. Differentiating (2.26) with respect to x, setting the result equal to zero and solving for x, proves the proposition. □

2.4.3 $P < \lambda$ demand is intermittent

In this case, the demand rate is higher than the production rate, but it is not constant. So, the production strategy shall be make-to-stock in order to catch up with the demand. We assume that the demand is known and the production is started soon enough to make sufficient stock in order to catch up with the demand. Figure 2.7 shows changes in the inventory of a SKU in this system. In period T_1, the SKU is produced and stored in the warehouse. No depletion occurs in this period and lanes are filled at the rate P. The depletion starts in period T_2, while the SKU is still produced and stored in the warehouse. In this period, the demand is fulfilled by the stock and production together. Since the demand rate is higher than the production rate, lanes are depleted at the rate $(\lambda - P)$ in T_2. An example of this case is seasonal products.

Inventory reaches its maximum level at the end of T_1; therefore, T_1 is (H/P). The entire production batch is depleted in T_2; hence, T_2 is (Q/λ) and the inventory cycle time, T, is equal to (Q/P). The maximum number of lanes required for storage is $(\lceil H/zx \rceil)$ where H is $(Q - \lfloor QP/\lambda \rfloor)$. Relaxing the integrality restriction results in the following approximations:

$$H \approx \frac{Q\,(\lambda - P)}{\lambda}, \tag{2.29}$$

$$K \approx \frac{Q\,(\lambda - P)}{\lambda z x}. \tag{2.30}$$

One should notice that (2.29) could also be obtained by substituting the values of T, T_1 and T_2 in $T = T_1 + T_2$.

2.4.3.1 Optimal lane depth for a single SKU

W_H in T_1 and T_2 is calculated by replacing filling rates in (2.17) with P and depletion rates in (2.3) with $(\lambda - P)$. W_A and O_T are similarly calculated by updating the new filling and depletion rates in (2.21) and (2.22), respectively. It follows that

$$U \approx \frac{2x(Q(\lambda - P) + 2P - \lambda)}{(2x+a)(Q(\lambda - P) + \lambda z x + 2P - 2\lambda)}. \tag{2.31}$$

W is determined by summing all types of waste and then dividing the result by T, which is (Q/P). That is,

$$W \approx \left(\frac{1}{4\lambda x}\right)\left(2\lambda x(zx-1) + a\lambda(Q + zx - 2) - aP(Q-2)\right). \tag{2.32}$$

Proposition 2.5. *The optimal lane depth to block-stack a SKU whose batch of pallets is produced at the rate P and depleted at the rate λ, where $P < \lambda$, is*

$$x^* \approx \sqrt{\frac{a(Q-2)(\lambda - P)}{2z\lambda}}. \tag{2.33}$$

Proof. It is proven by taking the derivative of (2.32) with respect to x, setting it equal to zero and solving for x. □

2.4.3.2 Common optimal lane depth for multiple SKUs

W_U is calculated as described in section (2.4.2.2) using the new filling and depletion rates. W_A is computed as described for the single SKU model. Herein, the aisle space that a lane requires for accessibility changes to $(ez/2)$. To take the pallet heights into account, all waste expressions are scaled by multiplying by h_i. Summing the Ws for all SKUs results in the average waste of storage space for a common lane depth.

$$W_c \approx \sum_{i=1}^{n}\left(\frac{h_i}{4\lambda_i z_i x}\right)(\lambda_i(e_i(Q_i-2)(2x+a) + z_i x(2e_i x + ae_i - 2Q_i + 2)) - P_i(Q_i-2)(2x(e_i - z_i) + ae_i)). \tag{2.34}$$

Algorithm 2.1 Pseudo-code of the simulation.

```
Initialize parameters (P_i, λ_i, Q_i, z_i, x, a, n)
Schedule first events for all SKUs
while (simulation time not terminated) do
    Find the earliest event
    if (the event is production (arrival)) then
        if (a partially occupied lane is available for the SKU) then
            Assign the SKU to the lane
        else
            Dedicate a new lane to the SKU
    else
        if (a partially occupied lane is available for the SKU) then
            Deplete the SKU from the lane
        else
            Deplete the SKU from a fully occupied lane
    Update the lane occupancies
    Update W_H, W_A, W_U and O_T
    Schedule the next event for the SKU
return W_c and U_c
```

It follows that

$$U_c \approx \frac{2x \sum_{i=1}^{n} \left(\frac{h_i}{\lambda_i}\right) (Q_i(\lambda_i - P_i) + 2P_i - \lambda_i)}{(2x + a) \sum_{i=1}^{n} e_i h_i (x + \left(\frac{1}{\lambda_i z_i}\right) (Q_i(\lambda_i - P_i) + 2P_i - 2\lambda_i))}. \tag{2.35}$$

Proposition 2.6. *The optimal common lane depth to block-stack n SKUs whose batches of pallets are produced at the rate P and depleted at the rate λ, where $P < \lambda$, is*

$$x_c^* \approx \sqrt{\frac{a \sum_{i=1}^{n} \left(\frac{e_i h_i}{z_i \lambda_i}\right) (Q_i - 2)(\lambda_i - P_i)}{2 \sum_{i=1}^{n} e_i h_i}}. \tag{2.36}$$

Proof. Differentiating (2.34) with respect to x, setting the result equal to zero and solving for x, proves it. □

Table 2.2: Parameters of the triangular distributions used in the simulation.

Variables	Low	Mode	High
Production time	$\frac{1}{P}(1-0.3)$	$\frac{1}{P}$	$\frac{1}{P}(1+0.3)$
Demand inter-arrival time	$\frac{1}{\lambda}(1-0.5)$	$\frac{1}{\lambda}$	$\frac{1}{\lambda}(1+0.5)$
Batch size	$Q(1-0.3)$	Q	$Q(1+0.3)$

2.5 Experimental framework

The experimental framework is designed as follows: First, we describe the simulation model used to evaluate performance of the proposed models. Next, the test problem sets are described and then accuracy of the proposed models is evaluated by the simulation model. Finally, the finite and infinite production rate models are compared with respect to the optimal lane depth and space utilization.

2.5.1 Simulation model

The pseudo-code of the simulation model used for evaluation (an event-oriented model written in Python) is shown in Algorithm 2.1. The main goal of the experimental analysis is to evaluate accuracy of the proposed models for the real world situations where stochastic variations exist among the major production factors and demand. For this reason, the simulation model utilizes random variables for the production times, demand inter-arrival times, and the batch sizes as presented in Table 2.2.

To compute the performance metrics under the stochastic variations, the simulation model is run for lane depths from 5 to 50 pallets deep and replicated 40 times for each lane depth. The average waste of storage space is computed for each replication and the average of results across the replications is recorded for each lane depth. Finally the lane depth that generated the minimum average waste of space is reported as the optimal lane depth. Common random numbers were used for all lane depths in the same replication to ensure that randomness does not interfere in selecting the optimal lane depth. However, the number

Table 2.3: Parameters of the uniform distributions used to generate the random SKUs.

Variables	$P > \lambda$		$P < \lambda$	
	Min	Max	Min	Max
P (pallets/hour)	λ	100	0.5	10
λ (pallets/hour)	0.1	2	P	15
z (pallets)	2	5	2	5
a (pallets)	2	4	2	4
h (feet)	2	5	2	5
Pallet cost	50$	500$	50$	500$

of replications must be determined such that the confidence intervals on the average waste of space obtained for any two consecutive lane depths do not overlap. Our experiments showed that 40 replications sufficiently narrows the confidence intervals such that this condition is met. Also, they showed that the space utilization converges faster when the warm-up period is set to 10 percent of the simulation period.

Our objective is to evaluate the long time performance of the system, therefore the simulation period must be defined long enough to cover sufficient numbers of inventory cycles for each SKU in the test problems. Considering the randomly generated test problems, the inventory cycle times in our test problems vary from less than a month to higher than 6 months. Thus, we set the simulation period to five years for both single and the multiple SKU test problems.

2.5.2 Analyzing accuracy of the models

The performance of the models was analyzed on randomly generated test problems for single and multiple SKUs. The test problems were designed as described in the following:

- **Single SKU:** A repository of 1000 SKUs was randomly generated for each of the two finite production rate models. Uniform random distributions with the parameters shown in Table 2.3 were used to generate the SKUs. The following are the non-random parameters used in generating the SKUs:

- monthly holding cost: (pallet cost)×0.3/12
- setup cost: (pallet cost)×5
- warehouse clear height: 25 feet
- Q: computed by the EOQ model [19]

The SKU repository for the infinite production rate model was created by duplicating the SKU repository generated for the finite production rate with $P > \lambda$ and discarding the Ps.

The simulation model was run for the 1000 SKUs in each repository one by one and the results were compared with the outputs of the relevant models.

- **Multiple SKUs:** For each of the three cases investigated in this paper, three test sets were designed for 10, 50 and 100 SKU test problems. Each test set consists of 30 different test problems whose SKUs were randomly chosen from the relevant SKU repository. For all multiple SKU test problems, aisle width was set to three pallets.

The accuracy of the proposed models in estimating the optimal lane depth and space utilization was evaluated by calculating the Mean Absolute Percentage Error (MAPE) between the model estimations and the simulation results. The MAPE for the optimal lane depth is calculated as

$$MAPE_{x^*} = \frac{1}{N}\sum_{i=1}^{N}\frac{\left|x_i^S - x_i^*\right|}{x_i^S}, \tag{2.37}$$

where x_i^S and x_i^* are the optimal lane depth obtained by the simulation and the proposed models for the ith test problem, respectively, and N is the number of the test problems, which is equal to 1000 and 30 for the single and multiple SKU test sets, respectively. The MAPE for space utilization is obtained by

$$MAPE_U = \frac{1}{(N \times 45)}\sum_{i=1}^{N}\sum_{j=5}^{50}\frac{\left|\bar{U}_{ij}^S - U_{ij}\right|}{\bar{U}_{ij}^S}, \tag{2.38}$$

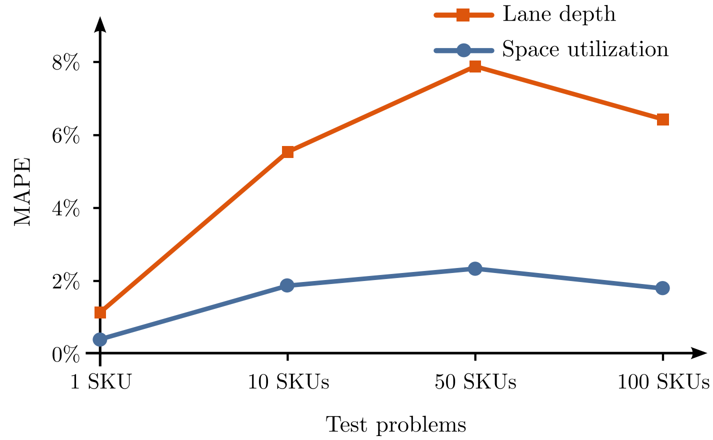

Figure 2.8: The MAPE for the optimal lane depth and space utilization, $P = \infty$.

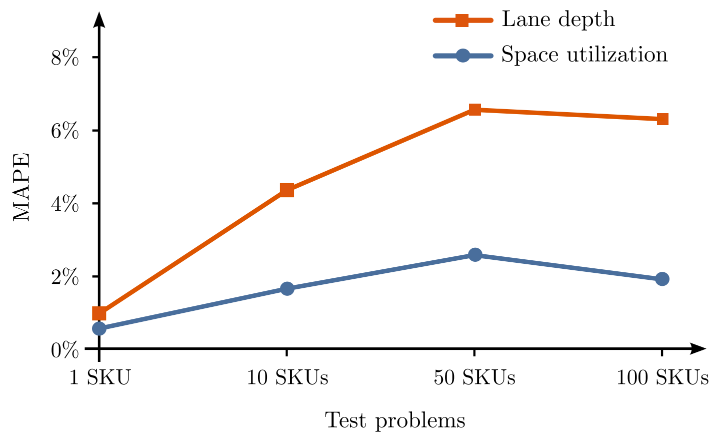

Figure 2.9: The MAPE for the optimal lane depth and space utilization, $P > \lambda$.

where $\bar{U}_{ij}^S$ is the average space utilization within the 40 replications of simulation for the lane depth j in test problem i and U_{ij} is the space utilization estimated by the relevant model for the lane depth j in test problem i. Figures 2.8, 2.9 and 2.10 show the $MAPE_{x^*}$ and $MAPE_U$ for the three investigated cases. The following observations are obtained from the experimental study:

- The $MAPE_U$ and $MAPE_{x^*}$ are less than 2.6 and 8 percent for all three cases for all four test sets, respectively. This shows that despite the high variations applied to the P, λ, and Q in the simulations, the models accurately estimated both the optimal lane

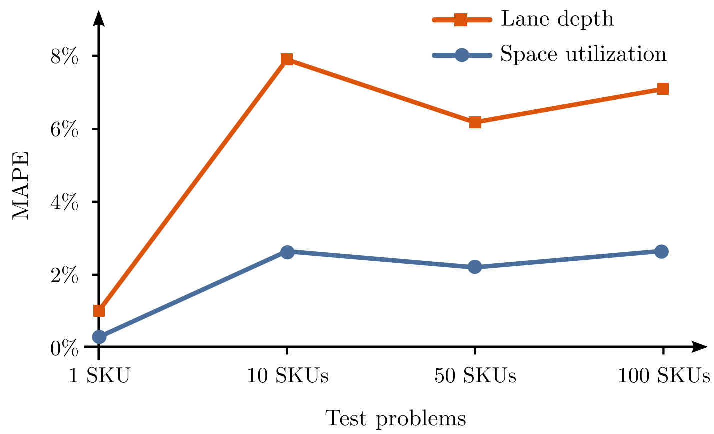

Figure 2.10: The MAPE for the optimal lane depth and space utilization, $P < \lambda$.

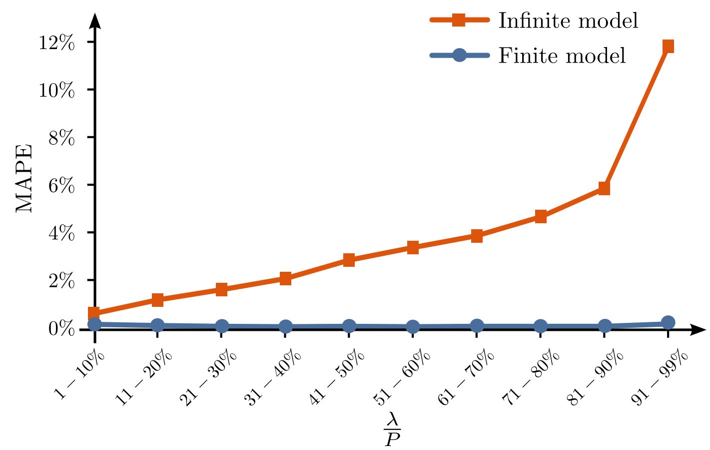

Figure 2.11: The MAPE for the space utilization estimated by the finite and infinite production rate models.

depth and space utilization in all three cases and their performance are robust under the presence of stochastic variations.

- Performance of the models are consistent and do not depend on the size of the problems (number of SKUs). The $MAPE_U$ varies less than one percent as the number of SKUs increases from 10 to 100 SKUs. This variation is less than three percent for the $MAPE_{x^*}$.

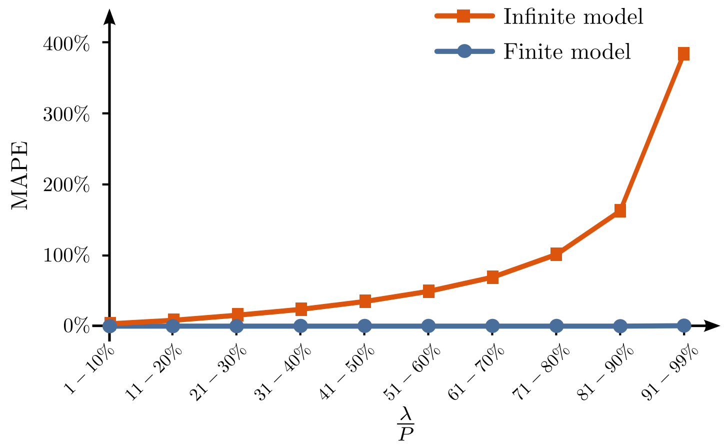

Figure 2.12: The MAPE for the optimal lane depth estimated by the finite and infinite production rate models.

- Single SKU models estimated both performance metrics more accurately than the multiple SKU models, though the difference is relatively small. The $MAPE_U$ and $MAPE_{x^*}$ are less than 0.5 and 1 percent for these models, respectively.

2.5.3 Finite vs. infinite production rate model

The infinite production rate model is the only existing model in the literature that addresses optimal lane depth in block stacking warehouses [18]. Employing this model in the warehouses located in manufacturing systems is equivalent to disregarding the production flow to the warehouse and assuming instantaneous SKU arrivals. In this section, we show the disadvantages of disregarding the production rates in such warehouses. We compare the infinite production rate model with the finite model on the same test problems using the simulation model to compute space utilization for the optimal lane depths obtained by the two models. The experimental study in this section is designed as follows:

First, we test the single SKU models for different (λ/P) ratios. From a mathematical point of view, the infinite production rate model is a special case of the finite model where $P > \lambda$. That is, (2.10) can be alternatively obtained by substituting $(P = \infty)$ in (2.25). This experiment aims to determine when the infinite model is capable of estimating optimal lane depth for a finite production rate problem with a relatively small error. Then, performance of both models are examined for multiple SKUs with respect to space utilization. Single and multiple SKU test problems are designed for this experiment as described in the following:

- **Single SKU:** 10 different test problems each consisting of 1000 randomly generated SKUs were generated similar to the SKU repositories in section 2.5.2 except that here, the demand rates were generated such that the (λ/P) ratios for all SKUs in the first, second, ..., and the tenth test problems were in the ranges 1-10 percent, 11-20 percent,..., 91-99 percent, respectively.

- **Multiple SKUs:** Two SKU repositories were randomly generated as described in section 2.5.2. Herein, the demand rates were generated such that (λ/P) ratios were between 5 to 30 percent for the SKUs in the first repository and between 70 to 95 percent for the ones in the second repository. Then, three test sets were designed each containing 30 test problems for 10, 50 and 100 SKUs, respectively. For each test problem, 70 percent of its SKUs were randomly chosen from the first SKU repository and the remaining were randomly chosen from the second repository. This setup aims to preserve diversity among the SKUs in the test problems.

Figure 2.11 and 2.12 compare the $MAPE_U$ and $MAPE_{x^*}$ for the single SKU models for different (λ/P) ratios. The infinite production rate model obtained relatively small errors when the ratio is less than 10 percent. As the ratio increases, both the $MAPE_U$ and the $MAPE_{x^*}$ drastically increase for the infinite model. On the contrary, the finite model obtained relatively small and consistent errors for all ranges.

Table 2.4 shows the test results for the multiple SKU test problems. On average, the optimal lane depths obtained by the finite production rate model resulted in about two percent higher space utilization than the ones obtained by the infinite model. This improvement is almost consistent among the three test sets. On the other hand, the infinite model obtained much deeper lanes than the finite model. The average of the optimal lane depths estimated by the infinite model is more than 68 percent deeper than the average

Table 2.4: Finite vs. infinite production rate models.

Metric	10 SKUs			50 SKUs			100 SKUs		
	Min	Max	Avg.	Min	Max	Avg.	Min	Max	Avg.
Improvement in space utilization by finite model	0.36%	3.67%	2.09%	0.87%	3.25%	1.94%	0.76%	2.68%	1.86%
Optimal lane depth (finite model)	14	21	17.8	15	19	17.60	16	19	17.43
Optimal lane depth (infinite model)	24	40	30.00	26	34	29.90	26	33	29.63

of the ones obtained by the finite model. Deep lanes form a warehouse layout that has few cross aisles and therefore, the transportation costs increase in the warehouse. So, the layout designed by the finite model is more flexible and also incurs less transportation costs. However, quantifying this cost is out of the scope of this paper and is an open problem for a future research.

The optimal lane depths obtained by the finite model achieved higher space utilization in all test problems. This improvement increases when the production rates are closer to the demand rates, like just-in-time manufacturing systems where the production lines are in balance with the demand.

2.6 Conclusions

In this paper, we developed mathematical models to obtain the optimal lane depth for single and multiple SKUs in block stacking storage systems under finite production rate constraints. The following cases were studied: infinite production rate, finite production rate where the production rate is higher than the demand rate, and less than the demand rate. A simulation model was developed to evaluate performance of the models for the real word situation where uncertainty exists in production and demand. That is, the assumptions of constant, deterministic production and demand were relaxed in the simulation. An experimental study was carried out on the randomly generated test problems for single and multiple SKUs. The experimental analysis shows that although the models were developed based on the assumption of deterministic production and demand, they are robust and accurate under uncertainty.

We evaluated the finite production rate model with the only existing model in the literature (infinite production rate model) on the same test problems and highlighted the advantages achieved in the space utilization and transportation costs by taking the production rates into consideration. The optimal lane depth obtained by the finite model led to higher space utilization in all test problems. They are nearly half as deep as the ones obtained by the infinite production rate model. This implies, they form a flexible layout that contains more cross aisles and as a result less transportation costs are incurred to the system.

Our proposed models accurately estimate the lane depth that enhances space utilization in block stacking warehouses in manufacturing systems. However, it is important to note that our model does not consider safety stock or transportation costs which could influence the results in practice. Considering both space utilization and Transportation costs in finding the optimal lane depth seems a substantial problem for future research.

Bibliography

[1] J. Ashayeri, L. Gelders, and L. V. Wassenhove. A microcomputer-based optimization model for the design of automated warehouses. *International Journal of Production Research*, 23(4):825–839, 1985.

[2] P. Baker and M. Canessa. Warehouse design: A structured approach. *European Journal of Operational Research*, 193(2):425–436, 2009.

[3] J. J. Bartholdi. A note on the most space-effcient lane-depth for block-stacked pallets. *Private Note*, 2014.

[4] H. J. Carlo, I. F. Vis, and K. J. Roodbergen. Storage yard operations in container terminals: Literature overview, trends, and research directions. *European Journal of Operational Research*, 235(2):412–430, 2014.

[5] M. Goetschalckx and H. Donald Ratliff. Optimal lane depths for single and multiple products in block stacking storage systems. *IIE Transactions*, 23(3):245–258, 1991.

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

[7] J. Gu, M. Goetschalckx, and L. F. McGinnis. Research on warehouse design and performance evaluation: A comprehensive review. *European Journal of Operational Research*, 203(3):539–549, 2010.

[8] K. R. Gue. Very high density storage systems. *IIE Transactions*, 38(1):79–90, 2006.

[9] K. R. Gue and R. D. Meller. Aisle configurations for unit-load warehouses. *IIE Transactions*, 41(3):171–182, 2009.

[10] K. R. Gue, E. Akcali, A. Erera, B. Ferrell, and G. Forger. *The U.S. Roadmap for Material Handling & Logistics*. MHI, Charlotte, NC, 2014.

[11] S. S. Heragu, L. Du, R. J. Mantel, and P. C. Schuur. Mathematical model for warehouse design and product allocation. *International Journal of Production Research*, 43(2):327–338, 2005.

[12] B. Hudock. Warehouse space and layout planning. In J. A. Tompkins and J. D. Smith, editors, *The warehouse management handbook*. Tompkins press, 2nd edition, 1998.

[13] D.-W. Jang, S. W. Kim, and K. H. Kim. The optimization of mixed block stacking requiring relocations. *International Journal of Production Economics*, 143(2):256–262, 2013.

[14] D. Kind. Elements of space utilization. *Transportation and Distribution Management*, 15:29–34, 1975.

[15] A. Klose and A. Drexl. Facility location models for distribution system design. *European Journal of Operational Research*, 162(1):4–29, 2005.

[16] T. Larson, H. March, and A. Kusiak. A heuristic approach to warehouse layout with class-based storage. *IIE Transactions*, 29(4):337–348, 1997.

[17] W. Marsh. Elements of block storage design. *International Journal of Production Research*, 17(4):377–394, 1979.

[18] J. O. Matson. *The analysis of selected unit load storage systems*. PhD thesis, Georgia Institute of Technology, Atlanta, GA, 1982.

[19] S. Nahmias. *Production & operations analysis*. McGraw-Hill/Irwin, New York, 5th edition, 2005.

[20] O. Öztürkoğlu, K. R. Gue, and R. D. Meller. Optimal unit-load warehouse designs for single-command operations. *IIE Transactions*, 44(6):459–475, 2012.

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

[22] J. A. Pazour and R. D. Meller. An analytical model for A-frame system design. *IIE Transactions*, 43(10):739–752, 2011.

[23] M. E. Petering and M. I. Hussein. A new mixed integer program and extended look-ahead heuristic algorithm for the block relocation problem. *European Journal of Operational Research*, 231(1):120–130, 2013.

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

[25] F. Ramtin and J. A. Pazour. Analytical models for an automated storage and retrieval system with multiple in-the-aisle pick positions. *IIE Transactions*, 46(9):968–986, 2014.

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

[27] M. J. Rosenblatt, Y. Roll, and D. V. Zyser. A combined optimization and simulation approach for designing automated storage/retrieval systems. *IIE Transactions*, 25(1): 40–50, 1993.

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

[29] J. Van den Berg. A literature survey on planning and control of warehousing systems. *IIE Transactions*, 31(8):751–762, 1999.

[30] J. Yang and K. Kim. A grouped storage method for minimizing relocations in block stacking systems. *Journal of Intelligent Manufacturing*, 17(4):453–463, 2006.

Chapter 3

Space-efficient layouts for block stacking warehouses

3.1 Abstract

In block stacking warehouses, pallets of stock keeping units are stacked on top of one another on the warehouse floor. The arrangement of lanes in the layout of this storage system affects utilization of the storage volume. Existing research that studies space utilization exclusively attempts to find the optimal lane depth and does not address the design of a space-efficient layout. We describe a model for block stacking warehouses that chooses a set of bay depths and arranges them in a layout to minimize wasted space. We use simulation to evaluate performance of the proposed model through an experimental analysis that covers small to industrial-sized warehouses.

Keywords: block stacking; facility design; layout design, warehouse design; space utilization

3.2 Introduction

Storing pallets of Stock Keeping Units (SKUs) on top of one another on a warehouse floor is known as block stacking. This inexpensive storage system does not generally require storage racks and can be inexpensively implemented in any open area. For this reason, it is widely used in manufacturing systems and distribution centers. Block stacking is mainly used with a *shared* or a *dedicated* storage policy. In the dedicated policy, lanes are dedicated to SKUs, and each SKU is allowed to be stored only in its assigned lanes, whereas in the shared (random) policy empty lanes are available to all SKUs. Hence, the shared policy is more efficient in utilizing storage space, but is generally less efficient for order picking. The space-efficient nature of this policy makes it a common space management policy for block stacking. However, to prevent lane blockage or extra pallet relocations, a lane in the shared policy is temporarily dedicated to the SKU that occupies its first pallet position, making unoccupied pallet positions of the lane unavailable to other SKUs. This effect is called *honeycombing* and waste associated with it remains in the system until the lane becomes fully occupied or empty.

As explained in Bartholdi and Hackman [2], aisles also contribute to the waste of space, because they are not used for pallet storage but are required to access lanes. To enhance utilization of the storage space, the warehouse must be designed such that both of these wastes are minimized. However, there is a trade-off. Layouts with shallow lanes generate less honeycombing waste but require more aisles, whereas the opposite is true for deep lanes.

Various studies have investigated layout design for conventional rack storage systems [1, 9]. These studies mostly considered designing the layout with respect to transportation costs for order picking [10, 11, 21, 22, 28, 3]. Further details can be found in [4]. Other research investigated this problem from the perspectives of operational cost [29, 20, 30], space utilization [7], product allocation [19, 25, 16], operating policies [24, 26, 12], and warehouse throughput [23, 14].

Kind [13] proposed the first model to take into account the trade-off between lane depth and width to find the lane depth that minimizes the waste of floor space. However, he did not provide any derivations for his formula. Later, Marsh [17] used simulation to evaluate space utilization on alternative lane depths and storage policies in this storage system.

Matson [18] extended Kind's model [13] and proposed a model to approximate the optimal lane depth when lanes are replenished instantaneously (i.e., replenishment rate is infinity). She also developed a model to find the common optimal lane depth for multiple SKUs. Her models are suitable for warehouses that store products received from suppliers, in which a truck unloads a batch of pallets at once (infinite replenishment rate).

Goetschalckx and Ratliff [8] showed that if multiple lane depths are allowed, then the optimal lane depths follow a triangular pattern. They developed a dynamic programming algorithm to select multiple optimal lane depths from a set of finite allowable lane depths so that the occupied floor space is minimized. They used a heuristic to form the warehouse layout by selecting at most five depths that form an approximately geometric series and then calculating the required number of lanes for each product based on the selected lane depths. The algorithm then rounds up or down the aggregated number of required lanes for each depth to the nearest multiple of lanes in an aisle.

Larson et al. [15] proposed a heuristic to design a class-based layout that maximizes floor space utilization and minimizes material handling cost. Their algorithm consists of three phases. In the first phase, the aisle directions (layout) and storage zone dimensions are determined. Then, storage types (rack storage or floor storage) are determined for all SKUs, and the required storage space for each storage type is calculated. Finally, the floor space is allocated for the storage zones (types) based on their types, required number of storage locations, and throughputs.

Derhami et al. [7] extended Matson's model [18] with two models to minimize waste of storage volume instead of floor space. They developed two finite production (replenishment) rate models: one for continuous demand less than the production rate, and the other for demand greater than the production rate. They showed that using an infinite production rate model in a finite production rate system results in lane depths about twice as deep as they should be. However, the resulting waste of volume is not significant because the space utilization curve, as a function of lane depth, is quite flat as the lane depth increases.

Existing research on space utilization in block stacking systems focuses exclusively on determining the optimal lane depth and does not provide any insights on designing the warehouse layout. Consider a traditional block stacking warehouse layout like the one in Figure 3.1b. To design a layout one must answer the following questions:

- How many aisles and bays should the layout have?

- How deep should the bays be?
- How to assign SKUs to lanes?

To the best of our knowledge, no analytical model exists to answer these design questions comprehensively. The optimal lane depth model cannot be used to design an optimal warehouse layout because the cost function is different. The optimal lane depth model [7, 18] computes waste of aisle space only for the period that a lane is occupied. Therefore, it treats aisle space as waste only when a lane is occupied and considers it available storage space otherwise. In the layout design problem, the space dedicated to aisles is always waste whether the adjoining lanes are occupied or not. Therefore, using the optimal lane depth model to design a layout underestimates aisle waste or, in other words, it assumes that storage lanes are immediately replenished as soon as they become empty.

Another reason the optimal lane depth model is inappropriate for the layout design problem is that it does not take warehouse dimensions into account. It attempts to optimize the trade-off between a block depth and width. Thus, it is a suitable tool to find the block size for temporary storage in a wide area.

In this research, we analyze waste of storage volume in block stacking from a layout design perspective. Our approach relaxes the immediate replenishment assumption of the optimal lane depth model and considers the volume dedicated to aisles as wasted storage volume for the entire planning horizon. We define a mixed integer programming model that finds the optimal number of bays and depths to minimize the total waste of the storage volume. The model allows multiple bay depths in the layout. We design an experimental study to investigate computational difficulty of the model and use a simulation model to evaluate performance of our model under stochastic conditions.

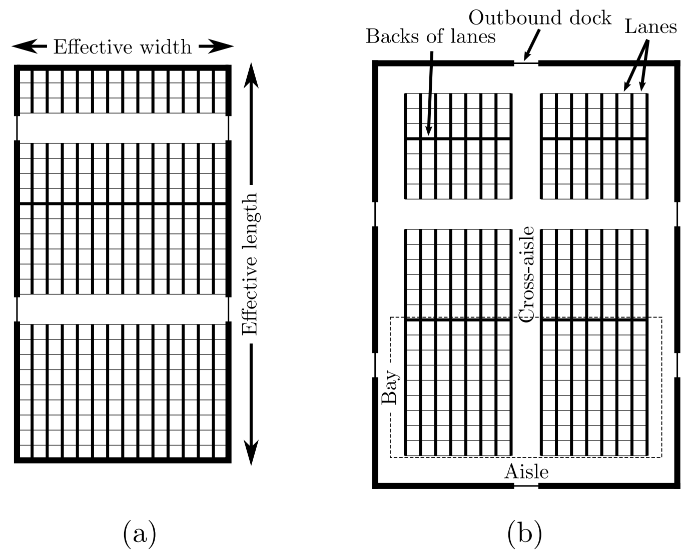

Figure 3.1: A layout generated by the proposed model (a) and after adding cross-aisles and an extra aisle (b).

3.3 Waste of the storage volume

The warehouse layout defines the shape, location and size of bays, lanes, aisles and cross-aisles on the warehouse floor. The layout is defined, in this research, as the number and arrangement of bays, aisles, cross-aisles, and bay depths for the given warehouse area. A typical warehouse layout and its elements are shown in Figure 3.1b.

Cross-aisles are used to facilitate access to lanes and to reduce travel distances inside the warehouse. So, unlike picking aisles, they are not necessary for pallet storage and consequently their space is considered pure waste of storage space. Because the objective of our model is to maximize utilization of the storage volume, including the number of cross-aisles as a decision variable in the model would lead to zero cross-aisles. Hence, they are not considered in the modeling, and we assume that the number of cross-aisles is given based on the warehouse width, material handling system, and traffic congestion [6]. To better utilize the storage volume, we assume that each aisle is shared between two bays. Therefore, any additional aisles and cross-aisles (for ease of transportation) must be added afterward (Figure 3.1).

We assume:

- The production schedule and production sequences are not known in advance.
- The warehouse is sufficiently large and the production sequences will be such that the warehouse can accommodate all produced SKUs under a shared storage policy.
- All lanes in the same bay have the same depth.
- Lanes are accessible from one side and they are depleted in the Last-In-First-Out (LIFO) order.
- Lanes are perpendicular to the short sides of the warehouse (labeled "Effective width" in Figure 3.1).
- The warehouse is unit-load (pallets).

To simplify modeling, we represent dimensions in units of floor space (pallets) rather than units of distance like feet or meters. In the next section, we calculate the total waste of storage volume and describe a model to minimize this waste. We develop the model for finite production rates where the production rates are bigger than the demand rates, and demand is continuous. Note that this model can be converted to instantaneous replenishment by letting the production rate equal infinity.

3.3.1 Waste of storage space

Assume that SKU i is produced in batches of Q_i pallets at rate P_i pallets per unit time and the pallets are stored on the floor at the same rate. Pallets are retrieved from the storage lanes at rate λ_i pallets per unit time, where $P_i > \lambda_i$. Assume that pallets of this SKU are H_i feet high and can be stacked up to Z_i pallets. The change in the inventory of this SKU over the cycle time is shown in Figure 3.2. The maximum inventory of the SKU is

$$V_i \approx \frac{Q_i\,(P_i - \lambda_i)}{P_i}, \tag{3.1}$$

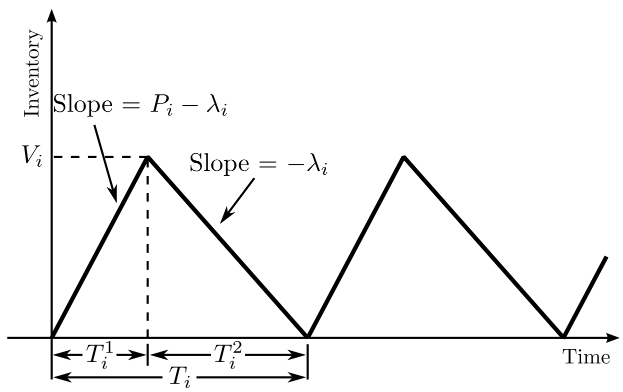

Figure 3.2: Changes in the inventory of SKU i over time, $P_i > \lambda_i$.

and the number of storage lanes that this SKU requires if it is stored in lanes x_i pallets deep is

$$K_i = \left\lceil \frac{Q_i\,(P_i - \lambda_i)}{P_i Z_i x_i} \right\rceil. \tag{3.2}$$

Relaxing the integrality assumption yields

$$K_i \approx \frac{Q_i\,(P_i - \lambda_i)}{P_i Z_i x_i}. \tag{3.3}$$

Equations (3.1) and (3.3) are approximations because we assume that lanes are filled in T_i^1 at rate $P_i - \lambda_i$ rather than being replenished and retrieved at rates P_i and λ_i at the same period. Three types of waste are generated in the warehouse:

1. **Honeycombing waste:** Pallet positions in a partially occupied lane that are unoccupied but unavailable to the other SKUs. This waste remains in the system until a lane becomes entirely occupied or empty.

2. **Unoccupied volume at the top of stacks:** Unoccupied space between the top of lanes and the clear height of the warehouse. This waste is incurred as a result of different pallet heights or technical restrictions on the stackable heights (like pallet weights, safety policies, etc.) that lead to various stack heights for different SKUs.

3. **Volume dedicated to the aisles:** The dedicated aisle volume is not directly used for pallet storage and therefore considered as waste of storage space. Unlike optimal lane depth models, for which this waste is considered as long as a lane is occupied, it is considered a permanent waste in the layout design problem.

3.3.1.1 Honeycombing waste

Assume SKU i is stored in an empty lane with depth x_i pallets. $Z_i x_i$ pallets of this SKU can be stored in the lane. When the first pallet is stored, the entire lane is dedicated to this SKU; therefore, the remaining $(Z_i x_i - 1)$ pallet positions in the lane become unavailable to other SKUs. This waste is incurred to the system until the next pallet is stored. For the sake of simplicity, we assume that pallet positions are replenished at rate $(P_i - \lambda_i)$. Therefore, $(Z_i x_i - 1)H_i$ pallet-feet are wasted for $(1/(P_i - \lambda_i))$ time periods. Then, the next pallet is stored in the lane and waste of storage volume decreases to $(Z_i x_i - 2)H_i$. The honeycombing waste continues to decrease until the lane becomes fully occupied. The total honeycombing waste incurred by a lane replenishment is

$$\left(\frac{H_i}{P_i - \lambda_i}\right)\left((Z_i x_i - 1) + (Z_i x_i - 2) + \cdots + (Z_i x_i - (Z_i x_i - 1))\right). \tag{3.4}$$

Similarly, honeycombing waste is generated when a lane is being emptied. When the first pallet is retrieved from a fully occupied lane, one pallet position becomes unoccupied for $(1/\lambda_i)$ time period. Then, the next pallet is retrieved and the lane will have two unoccupied pallet positions for the same amount of time. The total honeycombing waste incurred by a lane retrieval is

$$\left(\frac{H_i}{\lambda_i}\right)\left(1 + 2 + \cdots + (Z_i x_i - 1)\right). \tag{3.5}$$

The total honeycombing waste generated by replenishing and retrieving a batch of Q_i pallets is obtained by summing (3.4) and (3.5), and multiplying the result by K_i. It follows that

$$W_i^H \approx \left(\frac{1}{2\lambda_i}\right)\left(H_i Q_i\left(Z_i x_i - 1\right)\right). \tag{3.6}$$

3.3.1.2 Waste of unoccupied volume at the top of stacks

We define the clear height of the warehouse as the highest stackable height. So, unoccupied waste at the top of a stack is zero if a SKU can be stored to the maximum stackable height. Consequently, it will be removed from the model if all SKUs have the same stackable height. This waste is incurred to the system for the entire time that a lane is partially or fully occupied. Therefore, the time that lanes will be occupied must be calculated. Consider storing a batch of Q_i pallets. Note that V_i pallet positions are replenished and emptied in T_i^1. Once the first pallet is stored in the first lane, that lane will be occupied until the remaining $(V_i - 1)$ positions are filled. Pallet positions are replenished at rate $(P_i - \lambda_i)$, hence the first lane remains occupied for $(V_i - 1)/(P_i - \lambda_i)$ time period in T_i^1. The second lane is being used when the $Z_i x_i$ positions of the first lane are filled. It remains occupied until $(V_i - Z_i x_i - 1)$ remaining positions are replenished. The same process applies to the rest of lanes and the total time that all lanes are occupied in T_i^1 is obtained from

$$\left(\frac{1}{P_i - \lambda_i}\right)\left((V_i - 1) + (V_i - Z_i x_i - 1) + (V_i - 2Z_i x_i - 1) + \cdots + (V_i - K_i Z_i x_i)\right). \tag{3.7}$$

Now consider retrieving a batch of V_i pallets. The lane that is retrieved last remains occupied until the entire batch of V_i pallets are gone. Since retrievals occur at rate λ_i, this lane remains occupied for $(1/\lambda_i)(V_i)$ time period in T_i^2. The lane that is retrieved before this lane, remains occupied for $(1/\lambda_i)(V_i - Z_i x_i)$ time period, and so on. Thus, the total time that lanes are occupied in T_i^2 is given by

$$\left(\frac{1}{\lambda_i}\right)\left(V_i + (V_i - Z_i x_i) + (V_i - 2Z_i x_i) + \cdots + (V_i - K_i Z_i x_i)\right). \tag{3.8}$$

The total waste of unoccupied volume at the top of lanes is obtained by adding (3.7) and (3.8) and multiplying the result by the volume wasted at the top of a lane, which is $(S^h - Z_i H_i)x_i$, where S^h is the clear height of the warehouse in units of distance. That is,

$$W_i^U \approx \left(\frac{Q_i(S^h - Z_i H_i)}{2P_i \lambda_i Z_i}\right)\left(Q_i(P_i - \lambda_i) + P_i Z_i x_i - 2\lambda_i\right). \tag{3.9}$$

3.3.1.3 Waste of the dedicated volume to the aisles

As opposed to the optimal lane depth models in which the waste of the dedicated volume to aisles is computed only for the period that a lane is occupied [7], we consider the volume devoted to aisles as a permanent waste. Thus, the total waste of the dedicated volume to the aisles is

$$W^A = AS^h S^w n, \tag{3.10}$$

where n is the number of aisles in the layout, A is the aisle widths, and S^w is the warehouse width.

3.3.1.4 Total waste of the storage volume in the warehouse

The total waste of storage volume in the warehouse is the sum of honeycombing waste, unoccupied volume at the top of stacks, and the volume dedicated to the aisles. Denote the least common multiple of the cycle times of all SKUs by T^L. It also can be considered as a long period of time (steady state) in which all SKUs will have sufficient inventory turns. Given that the cycle time of SKU i is Q_i/λ_i, the number of inventory turns for this SKU in T^L is $T^L\lambda_i/Q_i$. Therefore, the total W_i^H and W_i^U generated by SKU i in T^L is

$$W_i^{HU} = \frac{T^L \lambda_i (W_i^H + W_i^U)}{Q_i}. \tag{3.11}$$

The total waste in the warehouse is given by summing W_i^{HU} for all SKUs and adding the aisle volume to the result. Note that the aisle volume remains as a waste for the entire

T^L. Hence, the total storage volume wasted in the warehouse in T^L is given by

$$W = T^L A S^h S^w n + \sum_{i \in I} \left(\frac{T^L \lambda_i (W_H^i + W_U^i)}{Q_i} \right), \tag{3.12}$$

where I is the set of all SKUs stored in the warehouse. Dividing (3.12) by T^L gives the average waste of storage volume in the warehouse. That is,

$$\begin{aligned} \bar{W} = {} & A S^h S^w n + \frac{S^h}{2} \sum_{i \in I} x_i \\ & + \sum_{i \in I} \left(\frac{1}{2 P_i Z_i} \right) \big((Q_i (S^h - Z_i H_i) - Z_i H_i)(P_i - \lambda_i) - \lambda_i (2 S^h - Z_i H_i) \big). \end{aligned} \tag{3.13}$$

Expression (3.13) depends on the following decision variables: set of SKUs' assigned lane depths, x_i, and the number of aisles in the layout, n. It is also restricted to the following constraint: the sum of bay depths and aisle widths must add up to the warehouse length. This causes a trade-off between bay depths and the number of aisles. A layout with deep bays has fewer aisles but generates higher honeycombing waste while the reverse is true for a layout with shallow bays. In the next section, we develop a model to optimally address this trade-off and minimize the total waste of storage volume in the warehouse.

3.3.2 Designing the warehouse layout

We minimize (3.13) with a mixed integer programming model that finds the optimal values for n and x_i. We call it MBD to distinguish it from the common lane depth model, which we call CLD. The MBD finds the optimal number of aisles and also bay depths for the given warehouse dimensions. It allows multiple depths in a layout. Hence, to measure the waste of space, it assigns SKUs to bays considering the number of lanes that they require with respect to their assigned bay depths.

Since the warehouse is intended to operate under the shared policy, it is not spacious enough to dedicate the required number of lanes to all SKUs (i.e. the warehouse cannot accommodate all SKUs at the same time to their maximum inventory levels). Notice that this would not be an issue in a layout operated with a dedicated policy. Therefore, the model hypothetically expands the layout width to provide sufficient space to dedicate required number of lanes to all SKUs (i.e. sufficient hypothetical space to allow switching the operating policy to the dedicated policy). That is, the same number of hypothetical lanes are added to all bays.

The model then assigns SKUs to bays ensuring that all SKUs have been assigned to the exact number of lanes that they require with respect to their assigned bay depths, and no bay is over-assigned. The resulting SKU assignment is a space-efficient operating policy used to prioritize bays with empty lanes for assignment to incoming SKUs.

To provide a clear view to the readers, we first present the initial version of the MBD, which is a nonlinear model. The linearized model will be described next.

3.3.2.1 Nonlinear model

Sets and data:

B set of bays, $B = \{1, \ldots, b_{max}\}$

E expansion ratio

S^l warehouse length (in units of pallets)

S^w warehouse width (in units of pallets)

b_{max} maximum number of bays in a layout

Decision variables:

y_{ib} 1 if SKU i is assigned to bay b, 0 otherwise

r_b depth of bay b (in units of pallets)

e_b 1 if bay b exists in the optimal layout (i.e., $r_b > 0$), 0 otherwise

$$\text{Minimize} \quad AS^w \sum_{b \in B} e_b + \sum_{i \in I} \sum_{b \in B} y_{ib} r_b \tag{3.14}$$

Subject to

$$e_{b-1} = e_b \qquad \forall b \in \{2, 4, \ldots, b_{max}\} \tag{3.15}$$

$$\sum_{b \in B} r_b + \frac{A}{2} \sum_{b \in B} e_b = S^l \tag{3.16}$$

$$L^{min} e_b \leq r_b \leq L^{max} e_b \qquad \forall b \in B \tag{3.17}$$

$$\left\lceil \frac{Q_i \left(P_i - \lambda_i\right)}{P_i Z_i r_b} \right\rceil y_{ib} \leq S^w e_b \qquad \forall i \in I, b \in B \tag{3.18}$$

$$S^w e_b \leq \sum_{i \in I} \left\lceil \frac{Q_i \left(P_i - \lambda_i\right)}{P_i Z_i r_b} \right\rceil y_{ib} \leq ES^w \qquad \forall b \in B \tag{3.19}$$

$$y_{ib} \in \{0, 1\} \qquad \forall i \in I, b \in B \tag{3.20}$$

$$e_b \in \{0, 1\} \qquad \forall b \in B \tag{3.21}$$

$$r_b \geq 0, integer \qquad \forall b \in B \tag{3.22}$$

The objective function (3.14) minimizes the total waste of storage volume in the warehouse. It takes into account the variable parts of (3.13). Note that considering constraint (3.15), $x_i = \sum_{b \in B} y_{ib} r_b$ and $n = \sum_{b \in B} e_b/2$. We removed the common factor $S^h/2$ from both terms in (3.14).

Constraint (3.15) guarantees that the number of bays is twice the number of aisles. It pairs the existence of two subsequent bays together and hence forces the total number of existing bays to be even. As a result, $\sum_{b \in B} e_b/2$ gives the number of aisles in the layout. Note that b_{max} must be even. Constraint (3.16) ensures that the sum of all bay depths and aisle widths adds up to the warehouse length.

Constraint (3.17) relates e_b and r_b together in addition to setting lower and upper bounds on r_b. If bay b exists in the solution (i.e. $e_b = 1$), r_b is forced to be between L_{min} and L_{max}. Otherwise it is forced to be zero. Note that b_{max} is sufficiently large to allow the model to select the optimal number of bays. Hence, the optimal solution may have fewer bays than

b_{max}. In this case, a zero bay depth implies that the respective bay does not exist in the optimal layout.

Constraint (3.18) ensures that the total number of assigned lanes from a bay to a SKU does not exceed the number of lanes in a bay. As shown in (3.2), $\lceil Q_i\,(P_i - \lambda_i)\,/P_i Z_i r_b \rceil$ is the maximum number of lanes that a SKU occupies if it is assigned to a bay whose depth is r_b pallets.

Constraint (3.19) restricts the SKU assignments and ensures the SKU assignments are feasible. It aims to balance the assignment loads among bays and to prevent accumulating SKUs in more favorable bays (shallow bays that incur less honeycombing waste). Without this constraint, an infeasible solution with only two bays and all SKUs assigned to the shallower bay would be found. The left hand side of constraint (3.19) ensures that all lanes of an existing bay are assigned to at least one SKU. The right hand side restricts the total number of assigned lanes of a bay to all SKUs to be fewer than or equal to the warehouse width (the original number of lanes in a bay) multiplied by the expansion ratio, E.

The model hypothetically expands the number of lanes in all bays to provide enough space to dedicate the maximum number of required lanes to all SKUs. E is the minimum ratio that satisfies this condition. It must be determined carefully because a large value of E allows the model to assign many SKUs to the shallow bays to minimize $\sum_{i \in I} x_i$ and a small E results in an infeasible solution. We compute E for b_{min} and b_{max} and then select the highest ratio. The procedure is as follows. First, a common bay depth is calculated for a layout with b_{min} bays:

$$\bar{x}_{min} = \left\lceil \frac{S^l - \left(\frac{1}{2}\right) b_{min} A}{b_{min}} \right\rceil . \tag{3.23}$$

Then, using (3.2), K_i for $x_i = \bar{x}_{min}$ is calculated. Summing K_is for all SKUs and dividing the result by the total number of lanes in the layout gives

$$E_{min} = \frac{1}{b_{min} S^w} \left(\sum_{i \in I} \left\lceil \frac{Q_i\,(P_i - \lambda_i)}{P_i Z_i \bar{x}_{min}} \right\rceil \right) . \tag{3.24}$$

Similarly, E_{max} is calculated for b_{max}. The expansion ratio is

$$E = Max\{E_{min}\ ,\ E_{max}\} + \epsilon, \tag{3.25}$$

where ϵ is a small number to compensate for the approximation error due to the common bay depth assumption. We set it equal to 0.05 in our experiment.

3.3.2.2 Linearized model

We linearize the model by introducing the following two sets of decision variables:

x_{id} 1 if SKU i is assigned to a bay whose depth is L_d pallets, 0 otherwise

z_{ib} number of lanes of bay b assigned to SKU i

The following set and data are used in the new model in addition to the ones defined before.

D set of allowable depths, $D = \{1, \ldots, d_{max}\}$

L_{min} minimum allowable bay depth

L_{max} maximum allowable bay depth

L_d dth allowable bay depth (in units of pallets), $L = \{L_{min},\ L_{min}+1\ , \ldots,\ L_{max}\}$

R_{id} number of lanes that SKU i occupies when stored in a bay whose depth is L_d pallets

S^l warehouse length (in units of pallets).

M_i arbitrarily large number, ($i = \{1, \ldots, 3\}$).

$$\text{Minimize} \quad AS^w \sum_{b \in B} e_b + \sum_{d \in D} L_d x_{id} \tag{3.26}$$

Subject to

$$\sum_{d \in D} x_{id} = 1 \qquad \forall i \in I \tag{3.27}$$

$$\sum_{b \in B} z_{ib} = \sum_{d \in D} R_{id} x_{id} \qquad \forall i \in I \tag{3.28}$$

$$z_{ib} \leq S^w e_b \qquad \forall i \in I, b \in B \tag{3.29}$$

$$S^w e_b \leq \sum_{i \in I} z_{ib} \leq ES^w \qquad \forall b \in B \tag{3.30}$$

$$y_{ib} \leq z_{ib} \leq M_1 y_{ib} \qquad \forall i \in I, b \in B \tag{3.31}$$

$$r_b \leq \sum_{d \in D} L_d x_{id} + M_2(1 - y_{ib}) \qquad \forall i \in I, b \in B \tag{3.32}$$

$$r_b \geq \sum_{d \in D} L_d x_{id} - M_3(1 - y_{ib}) \qquad \forall i \in I, b \in B \tag{3.33}$$

$$\sum_{b \in B} y_{ib} = 1 \qquad \forall i \in I \tag{3.34}$$

constraints (3.15)+(3.16)+(3.17)

$$y_{ib} \in \{0, 1\} \qquad \forall i \in I, b \in B \tag{3.35}$$

$$e_b \in \{0, 1\} \qquad \forall b \in B \tag{3.36}$$

$$r_b \geq 0, integer \qquad \forall b \in B \tag{3.37}$$

$$x_{id} \in \{0, 1\} \qquad \forall i \in I, d \in D \tag{3.38}$$

$$z_{ib} \geq 0, integer \qquad \forall i \in I, d \in D \tag{3.39}$$

The objective function (3.14) was linearized to (3.26) by introducing x_{id} and substituting $\sum_{i \in I} \sum_{b \in D} y_{ib} r_b$ with $\sum_{d \in D} L_d x_{id}$. Constraint (3.27) limits SKUs to only one depth. Constraints (3.28) and (3.29) linearize constraint (3.18). Constraint (3.28) forces the total number of assigned lanes to a SKU be exactly equal to the number of lanes that it requires with respect to the assigned bay depth. R_{id} is calculated using (3.2). That is,

$$R_{id} = \left\lceil \frac{Q_i (P_i - \lambda_i)}{P_i Z_i L_d} \right\rceil \qquad \forall d \in D. \tag{3.40}$$

Constraint (3.29) ensures that the total number of lanes assigned to a SKU from one bay does not exceed the warehouse width. Constraint (3.30) is a linearization of constraint (3.19). Constraint (3.31) relates z_{ib} and y_{ib}. The right hand side of (3.31) forces $y_{ib} = 1$ if a lane from bay b is assigned to SKU i (i.e., $z_{ib} > 0$). Otherwise, y_{ib} is forced to zero by the left hand side of 3.31). Since always $z_{ib} \leq S^w$, M_1 must be greater than or equal to S^w.

Constraints (3.32) and (3.33) assign bay depths to the SKUs. If SKU i is assigned to bay b (i.e., $y_{ib} = 1$), $1-y_{ib}$ becomes zero, forcing both constraints to work as an equality constraint that sets $\sum_{d \in D} L_d x_{id} = r_b$. Considering constraint (3.27), x_{id} will be one for $L_d = r_b$. If the SKU is not assigned to the bay, the right hand side of constraint (3.32) becomes a large positive number, making the constraint a loose constraint. Similarly, constraint (3.33) becomes non-binding as its right hand side becomes less than zero. M_2 and M_3 must be large enough to prevent violating these constraints when a SKU is not assigned to a bay. $M_2 \geq L_{max} - L_{min}$ and $M_3 \geq L_{max}$ satisfy this condition.

Constraint (3.34) ensures that each SKU is assigned to only one bay. It reduces the computational efforts of the model. Removing this constraint allows the model to assign lanes to a SKU from multiple bays that have the same depth. However, this increases the search space and also interferes with the cut that we will describe in section 3.3.3.3. If this constraint is included in the model, R_{i1} must be less than S^w for all SKUs; otherwise, the model will be infeasible. Those SKUs that do not satisfy this condition must be broken down into multiple SKUs with the same characteristics (P_i, λ_i, and Z_i) but smaller batch quantities, Q_i. The sum of the batch quantities of the child-SKUs must add up to the batch quantity of the original SKU and they all must meet this condition.

3.3.3 Solution techniques

The solution of the MBD model includes the optimal number of aisles, SKU assignments to bays and bay depths. The search space is highly symmetric as all possible combinations of bay depths with the same SKU assignments result in degenerate solutions that have the same objective value. We introduce symmetry-breaking constraints to remove these symmetric solutions from the search space. We also develop another class of inequalities to tighten the lower bound of the LP-relaxation and reduce the search space by developing tight lower and upper bounds on the number of bays.

3.3.3.1 Reducing problem symmetry

For every feasible solution in the MBD, there exist multiple degenerate solutions that have different depth orders but same depth assignments to the SKUs. For example, assume a warehouse with three bays and three SKUs. The following solutions all provide identical layout regarding space utilization.

$$\text{Sol 1: } r = (10, 15, 20)\ ,\ (y_{1,1} = 1,\ y_{2,2} = 1,\ y_{3,3} = 1)\ ,\ x = (10, 15, 20),$$

$$\text{Sol 2: } r = (15, 10, 20)\ ,\ (y_{1,2} = 1,\ y_{2,1} = 1,\ y_{3,3} = 1)\ ,\ x = (10, 15, 20),$$

$$\text{Sol 3: } r = (20, 10, 15)\ ,\ (y_{1,2} = 1,\ y_{2,3} = 1,\ y_{3,1} = 1)\ ,\ x = (10, 15, 20).$$

The following inequality prevents such symmetric solutions by forcing the bay depths to a non-decreasing lexicographic order.

$$r_b \leq r_{b+1} \qquad \forall b \in B - \{b_{max}\}. \tag{3.41}$$

This inequality forces the bay depths to a non-decreasing order and therefore arranges the empty bays (non-existing bays for which $r_b = 0$), if any exist, to the initial index values.

3.3.3.2 Tightening the LP-relaxation lower bounds

Set L contains the set of allowable depths bounded by L_{min} and L_{max}. Solving the model with many allowable depths increases the computational burden, so the bounds must be selected carefully. From the space utilization perspective, setting large L_{max} is preferable because it provides the model with more depth choices. However, retrieving and replenishing deep lanes are more laborious from the transportation and safety perspectives because forklifts have to travel longer distances inside narrow lanes. So, forklift restrictions, safety instructions, and other technical restrictions limit L_{max}. We arbitrarily set it equal to 30 pallets in our experiments.

Derhami et al. [7] showed that the space utilization curve, as a function of lane depth, drops significantly when lanes are too shallow. This loss is significant enough to prevent the model from selecting very small bay depths. So, we set L_{min} equal to 5 in our experiment. Therefore, $L = \{5,\ 6,\ ,\dots, 30\}$.

Setting tight lower and upper bounds on the number of bays significantly reduces the search space. L_{max} can be used to find a tight lower bound on the number of bays, as follows

$$b_{min} = 2\left\lceil \frac{S^l}{2L_{max} + A} \right\rceil. \tag{3.42}$$

L_{min} does not provide a tight upper bound on the number of bays because it is generally too small. We use the trade-off between honeycombing and accessibility waste to find b_{max}. As expression (3.6) shows, honeycombing waste depends on the lane depth. The deeper the lane, the more honeycombing waste is generated. It also depends on the frequency of retrievals and replenishments. The MBD aims to optimize the trade-off between honeycombing and accessibility wastes. If honeycombing waste is low, then the model makes bays deeper to decrease the number of aisles in the layout. But if honeycombing waste is significant, then the model minimizes the total waste by decreasing the bay depths and consequently increasing the number of aisles. Hence, the maximum possible honeycombing waste forces the maximum number of bays to the layout. For the sake of simplicity in modeling assume that all SKUs have the same stack height denoted by Z. The honeycombing waste generated to retrieve and replenish a lane with x pallets deep is obtained from

$$W_l^H = \left(\frac{1}{\lambda_i} + \frac{1}{P_i - \lambda_i}\right)\left(\frac{Zx(Zx-1)}{2}\right). \tag{3.43}$$

The time it takes to retrieve and replenish this lane is

$$t_l = \left(\frac{1}{\lambda_i} + \frac{1}{P_i - \lambda_i}\right) Zx. \tag{3.44}$$

The maximum honeycombing cost is generated in this lane when it never remains fully occupied or empty (it is replenished immediately after it becomes empty and emptied immediately after it becomes fully occupied). So, the highest honeycombing waste that this lane generates in T^L (long time) will be $W_l^H T^L/t_L$. It follows that

$$W_{l-max}^H = \left(\frac{1}{2}\right) T^L \left(Zx - 1\right). \tag{3.45}$$

Similarly, the maximum honeycombing waste in a bay with r_b pallets deep is obtained from

$$W_{b-max}^H = \left(\frac{1}{2}\right) T^L S^w \left(Zr_b - 1\right), \tag{3.46}$$

and the maximum honeycombing waste in the entire warehouse would be

$$W_{max}^H = \left(\frac{1}{2}\right) T^L S^w (Z \sum_{b \in B} r_b - b_{max}). \tag{3.47}$$

Adding the waste dedicated to the aisles, the total waste of storage volume in the warehouse is given by

$$W = \left(\frac{1}{2}\right) T^L S^w \left(Z \sum_{b \in B} r_b - b_{max} + ZAb_{max}\right). \tag{3.48}$$

Assuming common bay depth $\bar{x}$ for all bays yields

$$\sum_{b \in B} r_b = \bar{x} b_{max}, \tag{3.49}$$

and the total cost appears as

$$W = \left(\frac{1}{2}\right) T^L S^w b_{max} \left(Z\bar{x} - 1 + ZA\right), \tag{3.50}$$

subject to the following constraint:

$$b_{max}\left(\bar{x}+\frac{A}{2}\right)=S^{l}. \tag{3.51}$$

Solving (3.51) for $\bar{x}$ and substituting the result in (3.50) converts the total cost to a function of b_{max}. Taking the derivative of the new W with respect to b_{max}, setting it to zero, and then solving for b_{max} gives the optimal value for b_{max}:

$$b_{max}\approx\sqrt{\frac{S^{L}Z}{AZ-1}}. \tag{3.52}$$

Expression (3.52) provides a continuous approximation for b_{max}. The smallest even integer greater than or equal to b_{max} provides a tight upper bound on the number of bays. However, to compensate for the same bay depth and stackable height assumptions made in the modeling, we adjust b_{max} by incrementing it by a factor of two (the number of bays is even). In our experiment we increment it by two. The following inequality restricts the number of bays in the model:

$$b_{min}\leq n\leq b_{max} \tag{3.53}$$

Since bays are arranged in a nondecreasing order of their depths, the depth of the last b_{min} bays will be always nonzero. Hence, the following equality can be added to tighten the lower bound of the LP-relaxation:

$$e_b=1\quad\forall b\in\{b_{max}-b_{min}+1,...,b_{max}\}. \tag{3.54}$$

Furthermore, we propose to replace constraint (3.30) with the following constraint to tighten the lower bound of the LP-relaxation:

$$S^{w}e_b\leq\sum_{i\in I}z_{ib}\leq ES^{w}e_b\quad\forall b\in B. \tag{3.55}$$

Constraint (3.30) will be loose when bay depth is zero. This update changes it to a binding constraint when bay depth is zero.

3.3.3.3 Extra cut to reduce solution efforts

Derhami et al. [7] proposed the following formula to find the optimal lane depth for a single SKU:

$$x_i^* \approx \sqrt{\frac{A\left(Q_i(P_i - \lambda_i) - 2\lambda_i\right)}{2Z_iP_i}}. \tag{3.56}$$

Taking advantage of non-decreasing bay depths imposed by inequality (3.41), we use (3.56) to assign SKUs to bays based on the magnitudes of their optimal lane depths. We calculate x_i^* for all SKUs and sort them in a non-decreasing order. Let f_i be the index of the SKU located at the ith position of the sorted list. Then, the following inequality ensures that SKUs are assigned to bays based on the ascending order of their optimal lane depths:

$$\sum_{b \in B} b(y_{b,f_i}) \leq \sum_{b \in B} b(y_{b,f_{i+1}}) \quad \forall i \in \{1, \ldots, N_s - 1\}, \tag{3.57}$$

where N_s is the number of SKUs stored in the warehouse. Inequality (3.57) allows SKU i to be assigned to bay b only if all SKUs whose optimal lane depths are smaller than or equal to the optimal lane depth of SKU i are assigned to bays b and before. That means it assigns SKUs with smaller optimal lane depths to the shallower, initial bays. Although this inequality reduces the feasible region by restricting SKU assignments, it may remove some valid feasible integer solutions from the solution space. However, this may not considerably deteriorate the objective function as the order of the SKU assignment is still based on their optimal lane depths. We will study this cut from the computational perspective in the experimental analysis section and investigate its effect on the quality of the solution as well.

Table 3.1: Parameters of the uniform distributions used to generate the pool of random SKUs.

Parameter	Min	Max
P_i (pallets/month)	7200	36000
λ_i (pallets/month)	30	3000
Z_i (pallet)	2	4
H_i (feet)	3	5
c_i^p (dollars)	50	500

3.4 Experimental analysis

The experimental framework is as follows. First, the characteristics of the test problems are described in the next section. Then, computational difficulty of the model and effectiveness of the proposed cuts and bounds are analyzed on the test problems. Next, the simulation model used for the layout evaluation is described and finally the layouts obtained by the MBD model are evaluated by the simulation model and compared with the ones obtained by the CLD model.

3.4.1 Test problems

We generated test problems that vary from small to industrial-sized to analyze performance of the proposed model on different warehouse sizes. First, a pool of 4000 different SKUs was randomly generated. The parameters of the SKUs were sampled from the uniform random distributions whose parameters are shown in Table 3.1. c_i^p in the table is the cost of producing one pallet of SKU i. The Q_is were obtained using the Economic Order Quantity (EOQ) model as follows:

$$Q_i = \left\lceil \sqrt{\frac{2c_i^s \lambda_i}{c_i^h \left(1 - \frac{\lambda_i}{P_i}\right)}} \right\rceil, \tag{3.58}$$

where c_i^h is the monthly holding cost and was set to $c_i^p/4$, and c_i^s is the set-up cost to produce SKU i and was set to $5c_i^p$.

We designed 19 test problems with 10 to 1000 SKUs. SKUs in each test problem were randomly sampled from the pool of random SKUs. We used disproportionate stratified random sampling based on the SKUs' optimal lane depths. This is to ensure that multiple SKUs with a wide range of desirable lane depths exist in each test problem. The optimal lane depths were calculated using (3.56).

We divided the pool of SKUs into four groups such that the optimal lane depths for the SKUs in the groups were less than or equal to 13 pallets, between 14 and 18 pallets, between 19 and 24 pallets, and greater than or equal to 25 pallets, respectively 30% of the SKUs in each test problem were randomly selected from the first group, 35% from the second group, 20% from the third group, and 15% from the last group. We considered two cross-aisles (assuming one next to each long side of the warehouse) for the test problems that contain 50 SKUs or fewer and three cross-aisles (the additional one at the middle of the warehouse) for the remaining problems. The clear height of the warehouse was set to 16 feet for all test problems, and the aisle and cross-aisle widths were set to three pallets. We also assumed that pallet sizes are 42 by 42 inches.

Warehouse dimensions must be determined such that there would be sufficient space (storage and aisle) to accommodate the maximum possible inventory. To find the maximum possible inventory for each test problem, we developed an event-based simulation model only to keep track of the SKU inventories over the simulation time. We used the event log of the main simulation model for this purpose. We calculated the required floor space for the maximum inventory level recorded by simulation. Using (3.59), we then approximated the number of aisles in the warehouse for any given warehouse length. We assumed that the warehouse layout has a rectangular shape and its length is almost twice its width. Hence, we determined the warehouse length and width such that the available floor space for storage (warehouse area subtracted by the space dedicated to the aisles and cross-aisles) is 10% higher than the maximum required floor space (to account for the underestimated waste by the optimal lane depth model).

3.4.2 Computational experiment

The proposed model was coded with Python 2.7.11 and solved using Gurobi 6.0.5. The model was run on the Auburn University Hopper Cluster on Intel Xeon processors E5-2660 (2.6GHz) with 128 GB of RAM. We ran all experiments on 20 cores. We tested three scenarios to evaluate effectiveness of the proposed cuts and bounds:

- MBD, which only includes the MBD without any of the cuts and bounds developed in the paper.
- MBDC, which includes the MBD with the cuts (3.41), (3.53), (3.54), and (3.55) and excludes constraint (3.30).
- MBDCE, which includes the MBD with the cuts and the extra cut developed in the paper. It includes the MBD with cuts (3.41), (3.53), (3.54), (3.55), and (3.57) and excludes constraint (3.30).

To have a fair comparison, we disabled the built-in symmetry detection function in Gurobi but kept the other parameters of the solver to their default values. Also, a time limit of 10 hours was forced on the optimization process. Table 3.2 compares the computational efforts for all scenarios. As the results show, using the developed cuts and bounds reduces the solution time. The MBDC model found optimal solutions for the first two small test problems and feasible solutions within reasonable GAPs (up to 7.6% for the large problems) for the remaining problems that it did not solve optimally. While the MBD, which does not use any of the developed cuts and bounds, resulted in solving only one problem optimally in a significantly longer computational time (comparing to the MBDC) and no feasible solutions for the problems that contain 250 or more SKUs. Also, the MBDC model obtained smaller GAPs than the MBD model for all test problems (20-200 SKUs).

Comparing the MBDC with MBDCE shows that, as we expected, the extra constraint significantly improves performance. The MBDCE model found optimal solutions for the test problems that contain 150 or fewer SKUs and obtained best feasible solutions with relatively

Table 3.2: Computational efforts with/without the developed cuts and bounds.

Problems	MBD				MBDC				MBDCE			
	GAP	Obj.	Ex. nodes	Time	GAP	Obj.	Ex. nodes	Time	GAP	Obj.	Ex. nodes	Time
10 SKUs	0.00	11642.7	31811598	2421	0.00	11642.7	21934	2	0.00	11642.7	161	1
20 SKUs	5.30*	17268.6	179684803	36000	0.00	17268.6	239845458	22136	0.00	17276.6	9270	2
30 SKUs	9.25*	31259.9	96154756	36000	3.77*	31219.9	153487249	36000	0.00	31219.9	18915	17
40 SKUs	3.56*	35449.9	84169226	36000	2.37*	35329.9	72580693	36000	0.00	35353.9	30580	23
50 SKUs	3.86*	45108.3	71044945	36000	3.34*	45044.3	119435452	36000	0.00	45028.3	38253	31
100 SKUs	5.28*	92386.1	29715077	36000	5.01*	92282.1	62559424	36000	0.00	92194.1	331882	818
150 SKUs	3.77*	140945.8	15799077	36000	2.83*	140401.8	47609432	36000	0.00	140441.8	514334	2516
200 SKUs	5.74*	186478.3	736440	36000	5.48*	186318.3	1703775	36000	0.24*	185750.3	22742389	36000
250 SKUs	NIF	—	—	36000	5.37*	232791.4	25157771	36000	2.25*	232535.4	7277756	36000
300 SKUs	NIF	—	—	36000	4.32*	279102.9	19094755	36000	2.63*	279038.9	9910535	36000
350 SKUs	NIF	—	—	36000	4.64*	325601.5	7765123	36000	1.31*	324945.5	5070159	36000
400 SKUs	NIF	—	—	36000	7.50*	374733.0	5596467	36000	3.72*	369837.0	1477441	36000
450 SKUs	NIF	—	—	36000	5.05*	421758.6	8519992	36000	3.79*	421406.6	1638156	36000
500 SKUs	NIF	—	—	36000	5.32*	456491.3	8252609	36000	4.01*	455459.3	562694	36000
600 SKUs	NIF	—	—	36000	5.02*	531191.4	2917775	36000	NIF	—	—	36000
700 SKUs	NIF	—	—	36000	6.63*	650150.5	7047738	36000	3.76*	643254.5	—	36000
800 SKUs	NIF	—	—	36000	7.05*	732353.8	2289581	36000	NIF	—	—	36000
900 SKUs	NIF	—	—	36000	7.56*	822260.9	2494914	36000	NIF	—	—	36000
1000 SKUs	NIF	—	—	36000	7.04*	920743.6	1507563	36000	NIF	—	—	36000

*Optimization prematurely terminated after 10 hours of computation.
NIF: no integer solution found after 10 hours of computation.

small GAPs (up to 4.01%) for the remaining problems that it could not solve optimally. However, it did not find feasible solutions for the test problems with 600, 800, and more SKUs. This is because cut (3.57) forces additional limitation on the SKU assignments and adds extra complexity to the model as the problem size increases.

From the computational point of view, the MBDCE performed faster than MBDC. This is because it explored fewer nodes to find the optimal solutions. The objective values of the solutions obtained by the MBDCE are also very close to that of the MBDC. Both models obtained the same objective values for the 10 and 30 SKUs test problems. MBDCE did not perform as well as the MBDC model on the 20, 40, 150 SKUs test problems. Its solutions are 0.05%, 0.07%, and 0.03% higher than the MBDC, respectively. However, it improved the results of the MBDC between 0.02% to 1.31% for the remaining test problems. The small differences between the results of these two models show that the extra cut employed in the MBDCE model did not significantly deteriorate the objective values and led to small improvements in some cases.

From the computational perspective, the solution GAP increases, for all three models, as the problem size increases. Among the three models, however, the MBDCE model obtained smaller GAPs on the problems that it could solve (small to medium-sized problems). On the other hand, the MBDC model is capable of finding a feasible solution with relatively small GAP for the industrial-sized problems.

3.4.3 Analyzing performance of the layouts

We evaluated the layouts obtained by the MBD model using the simulation model of Derhami et al. [5]. They developed an event-oriented simulation model, coded in Python, to evaluate a given warehouse layout with respect to multiple performance metrics pertinent to space utilization and transportation cost. Their model simulates lane replenishment and retrieval operations under stochastic variations on the production times, demand, retrieval quantities, and production line set-up times. We tuned variations of these parameters in our experiments as follows. The production times were sampled from symmetric triangular distributions with parameters $(0.8/P_i, 1/P_i, 1.2/P_i)$ hour. Similarly, the outbound load times and production line set-up times were sampled from analogous distributions with parameters $(0.5/\lambda_i, 1/\lambda_i, 1.5/\lambda_i)$ hour and $(10, 20, 30)$ minutes, respectively. The retrieval quantities were sampled from a discrete uniform distribution with parameters $[1, 5]$ pallets.

We disabled the transportation module in the simulation because our analytical model does not take transportation into account. We set the warm-up period to one month, start-up inventories to zero, number of replications to 8, and simulation time to 8 months as described in [5]. We ran simulations on the Auburn University Hopper Cluster on Intel Xeon processors E5-2660 (2.6GHz) with 128 GB of RAM memory. We ran replications of the simulation on parallel processors and therefore used 8 cores for each experiment. We used common random numbers (CRN) across the replications for variance reduction. The following models were tested:

1. **CLD:** To have a baseline for performance comparison and also evaluate the layouts obtained by the common optimal lane depth model proposed in [7], we developed a simple algorithm to design the warehouse layout using the common optimal lane depth. The CLD algorithm works as follows. First, the common optimal lane depth is calculated as follows:

$$x_c^* = \left[\sqrt{\left(\frac{A}{2N_s}\right) \sum_{i \in I} \left(\frac{1}{Z_i P_i}\right) (Q_i(P_i - \lambda_i) - 2\lambda_i)} \right]. \tag{3.59}$$

   Then, the layout is divided into evenly deep bays whose depths are x_c^* pallets. Since this approach does not take the warehouse length into account, it is possible that the number of bays becomes an odd integer with the last bay depth smaller than x_c^*. This means one aisle is used to access only one bay instead of two. We remove this inefficiency by splitting the last bay to two equally deep bays if its depth is higher than 10 pallets. Otherwise, the last bay is equally split between the other bays and removed from the layout. The layouts produced by invoking this adjustment are marked in Table 3.5 for the respective test problems.

2. **MBDC-RAN:** The MBDC model finds the optimal bay depths and also SKU assignments. The optimal SKU assignment prioritizes bay assignment when more than one bay has empty lanes. That is, when a new pallet of a SKU requires a new empty lane, the storage lane is chosen among bays with empty lanes through the following process. First the assigned bay to the SKU is checked for any empty lanes. If no lane is available in that bay, bays whose depths are equal to the assigned depth to the SKU are checked for an empty lane. If such an empty lane is not available then the bay with the closest depth to the assigned depth to the SKU that has an empty lane is selected.

   It may be costly (time or labor) for non-automated material handling systems to follow the SKU assignments. Such warehouses may prefer employing a random SKU assignment rather than the optimal assignment. In the random assignment, an empty lane from a randomly selected bay is assigned to the incoming SKU. However, this decision imposes storage waste to the system. We are interested in first, examining the significance of the optimal SKU assignments on the volume utilization and second, estimating the loss in storage volume incurred by ignoring the optimal SKU assignments. For this reason, we tested the layouts obtained by the MBDC model under random SKU assignment policy and called it MBDC-RAN.

3. **MBDCE:** We showed that the MBDCE model is computationally faster than the MBDC model for small to medium-sized problems. However, this model removes some valid feasible solutions. We simulate the layouts obtained by this model to study the impact of the extra cut (3.57) on the quality of the solution.

4. **MBDC:** The layouts obtained by the MBDC model are compared, as baselines, with the ones obtained by the CLD, MBDC-RAN, and MBDCE.

Table 3.3 presents average waste of the storage volume (yd^3), volume utilization (%), and floor utilization (%) for the four scenarios. Interested readers are referred to [5] for more details on the definition and calculation of these parameters. We used a paired t-test to evaluate significant differences among the alternatives. Table 3.4 shows the test statistics and p-values for all comparisons. The paired t-test relies on the normality assumption among the pairs (replications). We used Shapiro-Wilk test [27] to examine normality of the differences between the pairs. The test statistics and p-values for the Shapiro-Wilk test are also shown in Table 3.4. The results of the normality tests show that the null hypothesis (i.e. samples are taken from a normal population) cannot be rejected for any of the comparisons at the significant level of 5%. The following alternatives are analyzed pairwise:

3.4.3.1 MBDC vs. CLD

The layouts produced by the CLD model imposed higher waste of storage volume than the MBDC's in all test problems. The paired t-test detects significant differences between

Table 3.3: Average waste of storage volume, volume utilization, and floor utilization obtained by simulation.

Problems	CLD			MBDC			MBDC-RAN			MBDCE		
	Waste	Vol. Util.	Fl. util.	Waste	Vol. Util.	Fl. util.	Waste	Vol. Util.	Fl. util.	Waste	Vol. Util.	Fl. util.
10 SKUs	10733±15	42.72±0.05	64.85±0.07	10674±16	42.85±0.05	65.05±0.07	10837±17	42.48±0.06	64.49±0.09	10674±16	42.85±0.05	65.05±0.07
20 SKUs	15282±13	49.51±0.04	68.96±0.04	14574±18	50.69±0.04	70.61±0.04	14836±12	50.25±0.04	69.99±0.05	14564±15	50.71±0.04	70.63±0.03
30 SKUs	23660±17	45.89±0.02	72.45±0.03	23526±18	46.03±0.02	72.67±0.02	23912±26	45.62±0.02	72.04±0.03	23532±19	46.02±0.01	72.66±0.02
40 SKUs	27407±16	51.26±0.01	74.01±0.02	26404±18	52.19±0.01	75.36±0.01	26946±18	51.68±0.01	74.63±0.02	26411±16	52.18±0.01	75.35±0.02
50 SKUs	33566±20	51.29±0.02	75.54±0.01	32573±15	52.04±0.02	76.65±0.02	33278±36	51.51±0.03	75.86±0.04	32504±19	52.10±0.02	76.72±0.01
100 SKUs	70617±40	50.02±0.02	76.05±0.02	69034±40	50.58±0.02	76.91±0.02	69648±40	50.36±0.02	76.58±0.02	68730±35	50.69±0.02	77.08±0.02
150 SKUs	98033±32	51.31±0.01	76.50±0.01	95514±24	51.96±0.01	77.47±0.01	96655±30	51.66±0.01	77.03±0.01	95118±26	52.06±0.01	77.62±0.01
200 SKUs	129212±37	52.02±0.01	78.27±0.01	125965±40	52.66±0.01	79.23±0.01	127823±54	52.29±0.01	78.68±0.01	125673±42	52.72±0.01	79.31±0.01
250 SKUs	155664±26	51.96±0.01	79.28±0.01	152994±24	52.39±0.00	79.94±0.00	156673±36	51.80±0.01	79.04±0.01	152842±23	52.42±0.00	79.98±0.01
300 SKUs	185261±33	52.47±0.01	79.86±0.00	181800±29	52.94±0.01	80.58±0.01	186578±67	52.30±0.01	79.59±0.01	181730±26	52.95±0.01	80.59±0.00
350 SKUs	212269±47	53.56±0.00	80.39±0.00	207845±39	54.09±0.00	81.17±0.00	211176±49	53.69±0.00	80.58±0.01	207531±44	54.12±0.00	81.23±0.00
400 SKUs	241280±42	53.52±0.01	80.32±0.00	236940±44	53.97±0.01	80.99±0.00	239286±27	53.73±0.01	80.62±0.01	234514±34	54.23±0.01	81.38±0.00
450 SKUs	272822±83	53.82±0.01	81.21±0.00	267527±82	54.30±0.01	81.95±0.00	273618±123	53.75±0.01	81.10±0.01	267348±74	54.32±0.00	81.97±0.01
500 SKUs	294926±33	54.82±0.00	81.27±0.00	287833±37	55.42±0.00	82.16±0.00	295697±56	54.75±0.01	81.17±0.01	287333±27	55.46±0.00	82.23±0.00
600 SKUs	339243±44	56.32±0.00	81.80±0.00	332589±47	56.81±0.00	82.51±0.00	339560±81	56.30±0.01	81.77±0.01	—	—	—
700 SKUs	387753±90	54.45±0.01	81.11±0.01	382820±72	54.76±0.01	81.59±0.01	387210±71	54.48±0.01	81.17±0.01	379576±78	54.97±0.01	81.90±0.00
800 SKUs	436859±48	55.31±0.00	81.08±0.00	428758±43	55.77±0.00	81.76±0.00	436664±111	55.32±0.01	81.10±0.01	—	—	—
900 SKUs	486855±64	55.28±0.00	81.30±0.00	479724±70	55.64±0.00	81.84±0.00	485179±100	55.37±0.00	81.43±0.00	—	—	—
1000 SKUs	536423±96	55.38±0.00	81.81±0.00	531729±86	55.60±0.00	82.13±0.00	541648±79	55.14±0.01	81.46±0.01	—	—	—

the two alternatives and rejects the null-hypothesis for all comparisons at the significant level of 5%. The layouts produced by the MBDC model generated, on average, 0.5% to 4.6% less waste of storage volume than the ones produced by the CLD model. This shows that, as it was explained in section 3.2, the CLD model underestimates the waste of storage volume (for the layout design purpose) and therefore, cannot find the optimal bay depths. However, the improvement in volume utilization is between 0.3% to 2.4%. This is because, as mentioned in [7], the volume utilization curve, as a function of bay depths, becomes flat around the optimal solution and therefore changes in the bay depth vector in the vicinity of the optimal solution do not result in significant changes in volume utilization.

3.4.3.2 MBDC vs. MBDC-RAN

This comparison shows the impact of using the optimal SKU assignment rather than the random assignment on the utilization of the storage volume. The layouts obtained by the MBDC model wasted less storage volume than the MBDC-RAN in all test problems. The paired t-test rejects equivalence of results between the two alternatives for all test problems. Discarding the optimal SKU assignments increased the average waste of storage volume between 0.9% to 2.7%. This shows that the optimal SKU assignments found by the MBDC model are statistically significant in better utilizing the storage volume.

Table 3.4: Statistical results of the the pairwise comparisons, $\alpha = 0.05$.

Problems	MBDC vs. CLD				MBDC vs. MBDC-RAN				MBDC vs. MBDCE			
	Paired t-test		Shapiro-Wilk		Paired t-test		Shapiro-Wilk		Paired t-test		Shapiro-Wilk	
	Stat.	p-value	Stat.	p-value	Stat.	p-value	Stat.	p-value	Stat.	p-value	Stat.	p-value
10 SKUs	47.95	4.49×10^{-10}	0.91	0.3766	30.4	1.07×10^{-08}	0.93	0.5532	—	—	—	—
20 SKUs	203.71	1.81×10^{-14}	0.92	0.4607	43.0	9.57×10^{-10}	0.93	0.5180	2.44	0.0447	0.91	0.3608
30 SKUs	65.37	5.15×10^{-11}	0.95	0.6670	74.4	2.08×10^{-11}	0.95	0.6772	4.34	0.0034	0.96	0.8247
40 SKUs	320.37	7.62×10^{-16}	0.87	0.1547	106.0	1.75×10^{-12}	0.94	0.6277	2.03	0.0814	0.91	0.3362
50 SKUs	323.77	7.08×10^{-16}	0.98	0.9811	52.9	2.25×10^{-10}	0.99	0.9986	32.77	6.37×10^{-09}	0.93	0.4792
100 SKUs	240.16	5.72×10^{-15}	0.90	0.3132	114.8	1.00×10^{-12}	0.86	0.1151	43.99	8.19×10^{-10}	0.96	0.8066
150 SKUs	427.89	1.00×10^{-16}	0.94	0.6390	105.6	1.80×10^{-12}	0.91	0.3742	83.89	9.00×10^{-12}	0.94	0.6506
200 SKUs	512.03	2.86×10^{-17}	0.89	0.2431	110.3	1.32×10^{-12}	0.91	0.3388	59.51	9.92×10^{-11}	0.84	0.0727
250 SKUs	391.05	1.88×10^{-16}	0.92	0.4178	172.4	5.84×10^{-14}	0.86	0.1280	40.91	1.35×10^{-09}	0.95	0.7459
300 SKUs	397.88	1.67×10^{-16}	0.87	0.1356	215.0	1.24×10^{-14}	0.86	0.1259	17.26	5.38×10^{-07}	0.84	0.0766
350 SKUs	446.36	7.48×10^{-17}	0.94	0.5812	217.2	1.15×10^{-14}	0.97	0.9022	42.56	1.03×10^{-09}	0.83	0.0533
400 SKUs	411.10	1.33×10^{-16}	0.96	0.8317	164.9	7.94×10^{-14}	0.97	0.9062	223.14	9.58×10^{-15}	0.95	0.7363
450 SKUs	531.50	2.20×10^{-17}	0.91	0.3855	215.3	1.23×10^{-14}	0.93	0.4770	20.44	1.68×10^{-07}	0.96	0.7963
500 SKUs	980.40	3.03×10^{-19}	0.89	0.2579	301.6	1.16×10^{-15}	0.83	0.0561	65.56	5.04×10^{-11}	0.93	0.5123
600 SKUs	778.94	1.51×10^{-18}	0.92	0.4067	310.9	9.41×10^{-16}	0.91	0.3790	—	—	—	—
700 SKUs	252.64	4.01×10^{-15}	0.94	0.6079	366.6	2.96×10^{-16}	0.98	0.9819	266.51	2.76×10^{-15}	0.91	0.3585
800 SKUs	696.36	3.32×10^{-18}	0.90	0.2833	187.0	3.29×10^{-14}	0.99	0.9985	—	—	—	—
900 SKUs	489.30	3.93×10^{-17}	0.92	0.4359	249.0	4.45×10^{-15}	0.88	0.1716	—	—	—	—
1000 SKUs	304.96	1.07×10^{-15}	0.86	0.1244	364.8	3.07×10^{-16}	0.91	0.3684	—	—	—	—

3.4.3.3 MBDC vs. MBDCE

Both models obtained the same solution for the 10 SKUs problem and hence the simulation results are the same for this test problem. The MBDC obtained slightly better solutions for 30 and 40 SKUs problems (0.03% improvement). For the remaining test problems, the MBDCE obtained slightly smaller waste of storage volume. It achieved 0.04% to 0.44% improvement over the MBDC results. The results of the paired t-tests show that they have explored significant differences between the two alternatives for all test problems. This shows that the extra cut (3.54) not only did not deteriorate the solution quality but it also slightly improved the quality of the feasible solutions obtained by the MBDC model.

3.4.3.4 Analysis of optimal layouts

The layouts obtained by the CLD, MBDC, and MBDCE models are displayed in Table 3.5. They are presented in pairs of a^b, which represent b bays with a pallets depth. Parameter β is the number of aisles to warehouse length ratio, which is calculated by dividing the number of aisles in the layout by the warehouse length.

Table 3.5: Layouts obtained by the proposed models (in units of pallets).

Problems	Warehouse dimensions		CLD				MBDC				MBDCE			
	Length	Width	Layout	# aisles	Avg. depth	β	Layout	# aisles	Avg. depth	β	Layout	# aisles	Avg. depth	β
10 SKUs	82	37	$(16^1, 20^3)$	2	19.0	0.024	$(12^2, 22^1, 30^1)$	2	19.0	0.024	$(12^2, 22^1, 30^1)$	2	19.0	0.024
20 SKUs	97	47	$(8^2, 18^4)$	3	14.7	0.031	$(14^1, 20^1, 27^1, 30^1)$	2	22.8	0.021	$(13^1, 22^1, 26^1, 30^1)$	2	22.8	0.021
30 SKUs	122	57	$(18^4, 19^2)^*$	3	18.3	0.025	$(12^1, 13^1, 15^1, 21^1, 26^2)$	3	18.8	0.025	$(13^1, 15^1, 16^1, 21^1, 22^1, 26^1)$	3	18.8	0.025
40 SKUs	132	62	$(6^2, 18^6)^*$	4	15.0	0.030	$(11^1, 17^1, 19^1, 20^1, 26^1, 30^1)$	3	20.5	0.023	$(13^1, 17^1, 19^1, 22^1, 23^1, 29^1)$	3	20.5	0.023
50 SKUs	142	72	$(4^1, 18^7)$	4	16.3	0.028	$(11^1, 16^1, 23^1, 27^1, 28^2)$	3	22.2	0.021	$(14^1, 18^1, 21^1, 23^1, 28^1, 29^1)$	3	22.2	0.021
100 SKUs	212	97	$(7^2, 18^{10})^*$	6	16.2	0.028	$(19^1, 24^2, 25^2, 27^1, 28^2)$	4	25.0	0.019	$(15^1, 20^1, 23^1, 25^1, 27^1, 30^3)$	4	25.0	0.019
150 SKUs	257	127	$(2^1, 18^{13})$	7	16.9	0.027	$(20^2, 21^3, 22^1, 28^1, 29^1, 30^2)$	5	24.2	0.019	$(11^1, 17^1, 21^1, 22^1, 26^2, 29^1, 30^3)$	5	24.2	0.019
200 SKUs	297	142	$(3^1, 18^{15})$	8	17.1	0.027	$(14^1, 20^1, 21^2, 22^1, 23^2, 24^2, 28^1, 29^1, 30^1)$	6	23.3	0.020	$(13^1, 16^1, 18^1, 20^1, 22^1, 23^1, 24^1, 25^1, 28^1, 30^3)$	6	23.3	0.020
250 SKUs	317	162	$(18^{14}, 19^2)^*$	8	18.1	0.025	$(12^1, 16^2, 24^2, 27^1, 30^6)$	6	24.9	0.019	$(13^1, 16^1, 19^1, 23^1, 25^1, 26^1, 27^1, 30^5)$	6	24.9	0.019
300 SKUs	357	172	$(18^{15}, 19^3)^*$	9	18.2	0.025	$(11^1, 12^1, 16^1, 17^1, 23^1, 25^1, 26^1, 28^1, 29^2, 30^4)$	7	24.0	0.020	$(12^1, 15^1, 17^1, 19^1, 21^1, 23^1, 24^1, 26^1, 29^1, 30^5)$	7	24.0	0.020
350 SKUs	377	187	$(5^1, 18^{19})$	10	17.4	0.027	$(14^1, 22^2, 23^2, 26^4, 29^2, 30^3)$	7	25.4	0.019	$(14^1, 17^1, 20^1, 21^1, 23^1, 25^1, 27^1, 29^1, 30^6)$	7	25.4	0.019
400 SKUs	407	202	$(7^2, 18^{20})^*$	11	17.0	0.027	$(24^4, 25^1, 26^1, 29^1, 30^7)$	7	27.6	0.017	$(13^1, 15^1, 17^1, 18^1, 21^2, 23^1, 24^1, 25^1, 26^1, 30^6)$	8	23.9	0.020
450 SKUs	427	217	$(16^1, 18^{21})$	11	17.9	0.026	$(14^1, 15^1, 16^1, 17^1, 19^1, 26^1, 28^1, 29^2, 30^7)$	8	25.2	0.019	$(13^1, 17^1, 20^1, 21^2, 23^1, 25^1, 26^1, 27^1, 30^7)$	8	25.2	0.019
500 SKUs	447	227	$(7^1, 8^1, 18^{22})^*$	12	17.1	0.027	$(9^1, 11^2, 15^2, 16^1, 23^1, 27^2, 29^4, 30^5)$	9	23.3	0.020	$(11^1, 14^1, 18^2, 20^3, 21^1, 25^3, 29^7)$	9	23.3	0.020
600 SKUs	497	242	$(8^1, 18^{25})$	13	17.6	0.026	$(15^1, 16^1, 17^1, 18^1, 25^3, 29^1, 30^{10})$	9	26.1	0.018	—	—	—	—
700 SKUs	527	262	$(8^1, 9^1, 18^{26})^*$	14	17.3	0.027	$(14^1, 26^2, 28^8, 30^7)$	9	27.8	0.017	$(16^1, 18^1, 20^3, 22^1, 23^3, 24^3, 30^8)$	10	24.9	0.019
800 SKUs	577	287	$(10^1, 18^{29})$	15	17.7	0.026	$(12^1, 21^1, 22^1, 23^1, 24^1, 26^1, 29^1, 30^{13})$	10	27.4	0.017	—	—	—	—
900 SKUs	617	297	$(11^1, 18^{31})$	16	17.8	0.026	$(14^1, 25^4, 26^4, 27^5, 28^3, 29^3, 30^2)$	11	26.5	0.018	—	—	—	—
1000 SKUs	627	317	$(18^{31}, 21^1)$	16	18.1	0.026	$(6^1, 13^1, 14^1, 21^1, 30^{18})$	11	27.0	0.018	—	—	—	—
Average	—	—	—	—	17.2	0.027	—	—	24.3	0.020	—	—	23.2	0.020

*Shows bays have been adjusted.

As Table 3.5 shows, CLD finds the same bay depth for almost all test problems, while the other two approaches use a diverse set of depths. The common lane depth is 20 pallets for the first problem and 18 pallets for the second problem. Then, it produces the same depth for the remaining problems.

The CLD model underestimates the optimal bay depth in almost all test problems. On average bay depths in the layouts produced by the CLD are 28% and 26% shallower than that of the MBDC and MBDCE models, respectively. This is because CLD underestimates the accessibility waste and consequently imposes unnecessary aisles in the layout.

The total average bay depth across the problems are 24.3 and 23.2 pallets for the MBDC and MBDCE models, respectively. Their layouts contain bays with 11 pallets and deeper (except two problems). This shows that honeycombing waste is not as costly as the accessibility waste; therefore, an optimal model tends to choose deeper bays to prevent having many aisles in the layout. This is because honeycombing waste is incurred only when lanes are being filled or depleted (i.e. no honeycombing waste when lanes are entirely occupied or emptied), and it also depends on the frequency of retrievals and replenishments, whereas the accessibility waste is permanent. Therefore, a layout with deep lanes better utilizes the storage space.

Using the β ratio, we suggest a rule of thumb to determine the number of aisles for a warehouse similar to our test problems. The β ratio equals 0.020 ± 0.001 for both MBDC and MBDCE models and remained almost steady for all test problems. Hence, a near optimal solution can be found by calculating the number of aisles by $n = \lceil \beta S^l \rceil$ and then dividing the layout into $2n$ evenly deep bays whose depths are $(S^l - An)/2n$ pallets.

3.5 Conclusions

We have developed a model to design a space-efficient layout for block stacking warehouses. We showed that the common lane depth (CLD) model is not appropriate to find bay depths for a layout and developed a new waste function to estimate the total waste of storage volume in the layout. We optimized this function with a mixed integer programming model to find the optimal bay depths and the number of aisles in the layout. We developed various cuts to reduce the problem symmetry and effectively bounded the decision variables to tighten the lower bound of the LP-relaxation (MBDC and MBDCE models). While both models produce good quality solutions, we found that the MBDCE model more likely performs better and faster on small to medium-sized problems (less than 700 SKUs) and the MBDC model performs better on large-sized problems.

The simulation experiments showed that the layouts produced by our models always generate less waste of storage volume than the layouts obtained by the CLD model. The improvement varied between 0.5% to 4.7% in our experimental study. We found that the CLD model underestimates the accessibility waste; as a result, its bay depths were on average 27% shallower than the optimal solution. In other words, the layouts obtained by the CLD model devoted up to 50% more space to the aisles. This model becomes insensitive as the number of SKUs increases, and the resulting bay depths remain steady between 18 to 20 pallets when more than 30 SKUs exist. However, the volume utilizations of the layouts produced by this model are close to the optimal solutions. This is because the volume utilization curve, as a function of bay depth, is almost flat around the optimal solution, and changes in bay depths would not result in significant changes in volume utilization. Hence, the CLD model produces near optimal solutions with respect to volume utilization.

We found that the accessibility waste outweighs honeycombing waste in their trade-off and an optimal model tends to choose deep bays for the layout. So, a layout with deep bays and fewer aisles utilizes the storage volume better than a layout with shallow bays. The average bay depth across the optimal solutions was 23.9 pallets in our experimental study. We introduced a new ratio, the number of aisles to warehouse length, to approximate the number of aisles for a given warehouse length. This ratio remained steady around 0.020 for all test problems in our experimentation. This can be used as a rule of thumb to find a near optimal solution for the warehouses similar to the ones tested in our experiments.

The results of our experiments also support that the optimal SKU assignments proposed by our model are statistically significant in better utilizing the storage volume, and implementing them decreased the wasted volume about 2% on average.

Bibliography

[1] P. Baker and M. Canessa. Warehouse design: A structured approach. *European Journal of Operational Research*, 193(2):425–436, 2009.

[2] J. J. Bartholdi and S. T. Hackman. *Warehouse & Distribution Science: Release 0.89*. Supply Chain and Logistics Institute, 2008.

[3] L. F. Cardona, D. F. Soto, L. Rivera, and H. J. Martínez. Detailed design of fishbone warehouse layouts with vertical travel. *International Journal of Production Economics*, 170, Part C:825–837, 2015.

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

[5] S. Derhami, J. S. Smith, and K. R. Gue. A simulation model to evaluate the layout for block stacking warehouses. *Progress in Material Handling Research: 2016*, 2016.

[6] S. Derhami, J. S. Smith, and K. R. Gue. Determining the optimal numbers of aisles and cross-aisles for block stacking warehouses: A simulation approach. *Manuscript in preparation*, 2017.

[7] S. Derhami, J. S. Smith, and K. R. Gue. Optimising space utilisation in block stacking warehouses. *International Journal of Production Research*, pages 1–17, 2017. doi: 10.1080/00207543.2016.1154216.

[8] M. Goetschalckx and D. H. Ratliff. Optimal lane depths for single and multiple products in block stacking storage systems. *IIE Transactions*, 23(3):245–258, 1991.

[9] J. Gu, M. Goetschalckx, and L. F. McGinnis. Research on warehouse design and performance evaluation: A comprehensive review. *European Journal of Operational Research*, 203(3):539–549, 2010.

[10] K. R. Gue and R. D. Meller. Aisle configurations for unit-load warehouses. *IIE Transactions*, 41(3):171–182, 2009.

[11] K. R. Gue, G. Ivanovć, and R. D. Meller. 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, 2012.

[12] X. Guo, Y. Yu, and R. B. de Koster. Impact of required storage space on storage policy performance in a unit-load warehouse. *International Journal of Production Research*, 54(8):2405–2418, 2016.

[13] D. Kind. Elements of space utilization. *Transportation and Distribution Management*, 15:29–34, 1975.

[14] T. Lamballais, D. Roy, and M. B. M. de Koster. Estimating performance in a robotic mobile fulfillment system. *European Journal of Operational Research*, 256(3):976–990, 2017.

[15] T. N. Larson, H. March, and A. Kusiak. A heuristic approach to warehouse layout with class-based storage. *IIE Transactions*, 29(4):337–348, 1997.

[16] J. Li, M. Moghaddam, and S. Y. Nof. Dynamic storage assignment with product affinity and ABC classification—a case study. *The International Journal of Advanced Manufacturing Technology*, 84(9):2179–2194, 2016.

[17] W. Marsh. Elements of block storage design. *International Journal of Production Research*, 17(4):377–394, 1979.

[18] J. O. Matson. *The analysis of selected unit load storage systems*. PhD thesis, Georgia Institute of Technology, Atlanta, GA, 1982.

[19] M. Moshref-Javadi and M. R. Lehto. Material handling improvement in warehouses by parts clustering. *International Journal of Production Research*, 54(14):4256–4271, 2016.

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

[21] O. Öztürkoğlu, K. R. Gue, and R. D. Meller. Optimal unit-load warehouse designs for single-command operations. *IIE Transactions*, 44(6):459–475, 2012.

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

[23] J. A. Pazour and R. D. Meller. An analytical model for A-frame system design. *IIE Transactions*, 43(10):739–752, 2011.

[24] F. Ramtin and J. A. Pazour. Analytical models for an automated storage and retrieval system with multiple in-the-aisle pick positions. *IIE Transactions*, 46(9):968–986, 2014.

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

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

[27] S. S. Shapiro and M. B. Wilk. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4):591–611, 1965.

[28] L. M. Thomas and R. D. Meller. Analytical models for warehouse configuration. *IIE Transactions*, 46(9):928–947, 2014.

[29] L. M. Thomas and R. D. Meller. Developing design guidelines for a case-picking warehouse. *International Journal of Production Economics*, 170, Part C:741–762, 2015.

[30] S. Zhou, Y. Y. Gong, and R. de Koster. Designing self-storage warehouses with customer choice. *International Journal of Production Research*, 54(10):3080–3104, 2016.

Chapter 4

Determining the optimal numbers of aisles and cross-aisles for block stacking warehouses: A simulation approach

4.1 Abstract

Storing pallets of products on the floor of a warehouse on top of one another is called block stacking. The arrangement of lanes, aisles, and cross-aisles affects utilization of the storage space and also transportation costs in this type of storage system. The existing literature focuses exclusively on determining the optimal lane depth to maximize space utilization and ignores transportation costs. In this paper, we develop a simulation-based optimization algorithm to find the optimal numbers of aisles and cross-aisles to maximize utilization of the storage space and to minimize transportation costs in a block stacking warehouse. Our computational experiments show that the proposed model finds the Pareto front in a reasonable time for large problems.

Keywords: block stacking; facility design; layout design, warehouse design; cross-aisle; transportation cost; space utilization

4.2 Introduction

Block stacking warehouses are unit load storage systems in which pallets of stock keeping units (SKUs) are stacked on top of one another on the warehouse floor. This type of storage system does not require storage racks and can be inexpensively implemented in any open area. However, space planning is challenging in this system. Block stacking is widely operated under the *shared* storage policy. In this policy, which is also known as *random* storage policy, unlike the dedicated storage policy, lanes are not dedicated to SKUs, and an empty

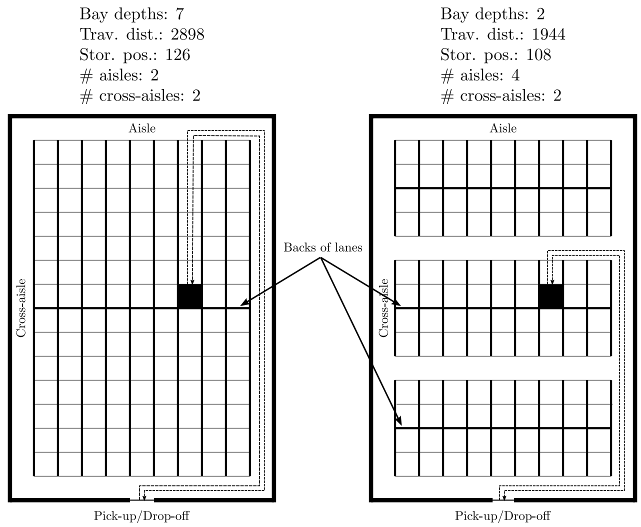

Figure 4.1: Travel distance vs. space utilization with respect to the number of aisles.

lane is available to all SKUs. However, to avoid blockage and relocation of pallets, a lane is temporarily allocated to a SKU once it occupies the first pallet position of the lane.

This restriction results in wasting storage space as there will be unoccupied pallet positions in a partially occupied lane that are not available to all SKUs. This effect is called *honeycombing* and waste associated with it is incurred by the system until a lane becomes entirely occupied or emptied [2]. Aisles also contribute to the overall amount of wasted space as they are used for accessibility rather than storage. There is a trade-off between honeycombing and accessibility wastes—shallow lanes generate less honeycombing waste but force more aisles to the layout while the opposite is true for deep lanes.

Transportation cost is another important factor that must be considered in designing a layout. The number of cross-aisles and their configurations (locations, directions, etc.) affect transportation costs within a warehouse, especially in warehouses operated with multi-command operations, where vehicles perform multiple replenishment or retrieval operations continuously before going to their home/parking. Cross-aisles decreases travel distances within the warehouse. However, like aisles, they are not used directly for pallet storage and subsequently, considered as waste of space.

Bay depths also affect the travel distances within a warehouse. Consider the two layouts presented in Figure 4.1. One layout has two bays each seven pallet positions deep, and the other one has the same dimensions but consists of six bays each two pallet positions deep. Note that the total warehouse area in both layouts are the same. Assume that aisles and cross-aisles are one pallet position wide. The total vertical distance a picker has to travel to and from the P/D point to replenish all pallet positions in both layouts are

Deep bays: $9((14 + 12+, \ldots, +2) + (44 + 42+, \ldots, +32)) = 2898$ floor-positions.

Shallow bays: $9((4 + 2) + 2(14 + 12) + 2(24 + 22) + (34 + 32)) = 1944$ floor-positions.

A longer distance must be traveled to replenish the layout with deeper bays. However, this layout has more storage positions than the other one. Note that the horizontal travel distances are the same for both cases, so they were not considered in the comparison. Derhami et al. [6] used simulation and showed there is a trade-off between space utilization and transportations costs with respect to bay depths. Hence, both of these objectives must be considered simultaneously to design a layout. Otherwise, part of the decision space is discarded and the layout performs poorly with respect to the ignored objective.

We develop an algorithm that aims to consider both space utilization and transportation costs in designing a warehouse layout. Our simulation-based algorithm optimizes the total travel distance within a warehouse by selecting the optimal number of cross-aisles and aisles while considering the space utilization as well. Our exhaustive experimental study shows that the algorithm finds the Pareto front for industrial-sized problems in a reasonable time.

4.3 Related research

The research studied the warehouse layout mostly considered the conventional warehouses with storage racks [30, 9, 1, 10]. These studies mostly aimed to design a layout that minimizes transportation costs for order picking [11, 12, 21, 22, 31, 3]. Interested readers are referred to [4] for further details on this research. The other objectives considered in designing a warehouse layout are operational costs [32, 20, 34], product allocation [19, 25, 17], warehouse throughput [23, 15], and operating policies [24, 29, 13].

Multiple articles exclusively studied the effect of cross-aisles on transportation costs. Roodbergen and de Koster [26] developed multiple heuristics to find the shortest path for order picking process when multiple cross-aisles exist in the layout. Vaughan and Petersen [33] proposed a heuristic to find the shortest path for order picking and studied the effect of adding cross-aisles on transportation costs. They showed that the number of cross-aisles that results in the maximum transportation efficiency depends primarily on the length of the storage aisles relative to the length of the cross-aisles. Roodbergen and Vis [27] proposed an analytical model to approximate the average length of an order picking route for two routing policies in a layout with one block (two cross-aisles). Their approximation can be used as an objective function in a nonlinear model to obtain the optimal number of aisles. Roodbergen et al. [28] developed an analogous approximation for a layout with multiple blocks.

Few published papers in the literature studied designing a layout for block stacking. Kind [14] considered the trade-off between the honeycombing and accessibility wastes to find the optimal lane depth. He proposed a model to approximate the lane depth that optimizes this trade-off. However, he did not provide any derivations for his model. Matson [18] developed another model to approximate the optimal lane depth under instantaneous replenishment (i.e., infinite storage rates). Her model was appropriate for warehouses that store products received from suppliers.

Goetschalckx and Ratliff [8] showed that if storage in multiple lane depths is allowed, the set of optimal lane depths follows a continuous triangular pattern. They developed a dynamic programming algorithm to select the set of optimal lane depths from a set of finite allowable lane depths to minimize the occupied floor space.

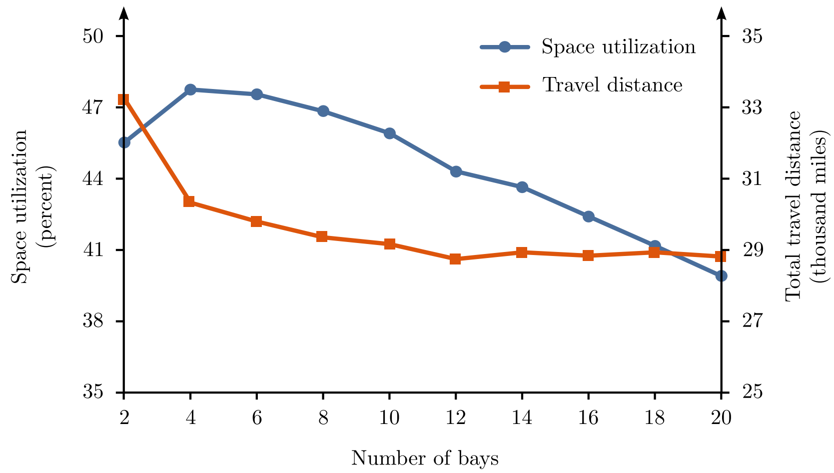

Figure 4.2: Space utilization vs. travel distance [6].

Larson et al. [16] proposed a heuristic to design a class-based layout. Their three-phases algorithm first characterizes aisle directions and dimensions of the storage zones. Then, it determines the storage types and assigns the required storage space to each storage zone.

Derhami et al. [5] further proposed a closed-form solution to find the optimal lane depth to maximize volume utilization under finite production rate constraint. They showed that using infinite production rate model in a finite production rate environment produces lane depths about twice as deep as they should be, but the resulting loss of space is not significant. This is because the space utilization curve, as a function of lane depth, becomes relatively flat as the lane depth increases. Derhami et al. [7] later showed that the optimal lane depth model [5, 18, 14] underestimates the accessibility cost for the layout design problem and developed an appropriate cost function to compute the total waste of storage volume in a layout with respect to the bay depths. They proposed a mixed integer programming model to optimize this function and to find the optimal bay depths.

Derhami et al. [6] used simulation to study space utilization and transportation costs in block stacking warehouses. Using a common bay depth, they simulated multiple layouts with various number of aisles to study the effect of bay depth on space utilization and travel distance. Figure 4.2 shows the results of their simulation for a warehouse with 50 SKUs. As the graph shows, utilization of the storage volume improves as the bay depth decreases.

It reaches its peak at the optimal bay depth and then starts decreasing as the bay depth becomes shallower. This is because the number of aisles (and consequently accessibility waste) increases in the layout as a result of having multiple shallow bays. On the other hand, the travel distance decreases as the bay depth decreases. However, this improvement becomes modest after a certain point. Hence, the transportation costs improve at the cost of lower space utilization.

Most of the research studied the design of block stacking systems focuses exclusively on improving utilization of the storage volume [18, 5, 7] and does not take the transportation costs into account. As discussed, both bay depths and the number of cross-aisles affect transportation costs. Hence, a comprehensive model should take into account the transportation costs as well as space utilization. Consider a conventional block stacking layout similar to the one presented in Figure 4.3. The following important design questions remain unanswered:

- How many cross-aisles should a layout have?
- How many aisles should a layout have to minimize transportation costs and maximize space utilization?
- How deep should bays be?

In this research, we develop a simulation-based optimization model to address these questions. Our model develops the warehouse layout considering two objectives simultaneously: maximizing space utilization and minimizing transportation costs. It takes into account the number of cross-aisles, their types, and the number of aisles as decision variables and uses simulation to evaluate different layouts with respect to these objectives. Using simulation enables the model to take the stochastic conditions that exist in the real world situation into account. It also provides the decision maker with a tool that accurately evaluates the trade-off between the two objectives in warehouses that use multi-command operations.

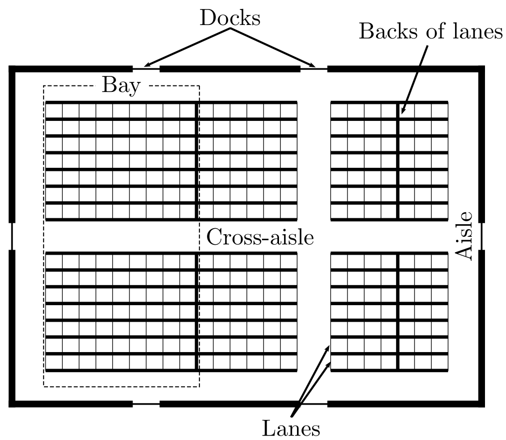

Figure 4.3: Components of a conventional block stacking layout.

4.4 Designing optimal layouts

The proposed model aims to find layouts that maximize space utilization and minimize transportation costs in a warehouse. The numbers of aisles and cross-aisles, their types (unidirectional vs. bidirectional), and their arrangements in the layout are the major design factors that affect these objectives. Cross-aisles decrease utilization of the storage space in a warehouse but improve transportation costs—especially in our targeted warehouse where vehicles continuously perform multiple retrieval or replenishment operations without returning to their home/parking. The number of aisles in the layout determines bay depths (assuming a common bay depth) and, as it was described before, there is a trade-off between the two objective functions with respect to bay depths.

Our model takes into account the number of aisles and cross-aisles and cross-aisle types as decision variables. To solve this multi-objective optimization problem, we develop a simulation-based optimization algorithm to explore the Pareto front and provide non-dominated solutions to the decision maker. The model consists of three main steps: layout generation, distance calculation, and simulation.

In the layout generation, layouts with different numbers of aisles and cross-aisles are generated. We develop a closed-form solution model to approximate the number of aisles that maximizes space utilization in the layout. This model along with numerical experiments

Algorithm 4.1 Pseudo-code of the proposed model

```
Import warehouse data
Calculate n* and determine n_min, n_max, c_min, and c_max
Generate all allowable layouts
for all (generated layouts) do
    Calculate distances considering unidirectional cross-aisles
    Calculate distances considering bidirectional cross-aisles
    Simulate the layout with the smaller total distance
Report non-dominated solutions
```

are used to determine lower and upper bounds on the number of allowable aisles and cross-aisles in a layout. The distances between all potential origins and destinations of a layout are calculated in the distance calculation step. Finally, the generated layout is evaluated by a steady state simulation. Algorithm 4.1 demonstrates the pseudo code of the proposed algorithm.

The following assumptions are made in this paper:

1. Lanes are arranged perpendicular to the short side of the warehouse.
2. All lanes in a bay have the same depth.
3. Cross-aisles are placed evenly spaced from each other. That is, the distance between any two subsequent cross-aisles is the same.
4. Unidirectional cross-aisles are one pallet position wide, and bidirectional cross-aisles are two pallet positions wide.
5. The unidirectional cross-aisles are placed in pairs of two unidirectional cross-aisles with opposite directions.
6. Aisles are bidirectional and two pallet positions wide (they must be wide enough to provide vehicles maneuvering space).
7. Traffic congestion in the warehouse is not directly modeled. Vehicles travel with a constant speed of V miles per hour. To accommodate for the lower speeds on the

turns, acceleration in straight lines, and delays and variations caused by the traffic congestion, we add stochastic variations to the calculated travel times.

8. Vehicles continuously perform multiple retrieval or replenishment operations per work shift without returning to their home/parking.

9. Aisles and cross-aisles next to the vehicle parking, pick-up points, and outbound docks are bidirectional.

10. Lanes are accessible from one side and they are emptied in the Last-In-First-Out (LIFO) order.

11. The number of bays is twice the number of aisles plus one. That is, each aisle is shared between two opposite bays, and the additional aisle is to keep two aisles next to the short side of the warehouse (see Figure 4.3).

12. The warehouse is unit-load (pallets).

The components of our algorithm are described in the following order. First, the layout generation algorithm is described in section 4.4.1. Then, the procedure to calculate travel distances in a layout is presented in section 4.4.2. After that, the simulation model is described in section 4.4.3. Finally, an experimental analysis is provided in section 4.5.

4.4.1 Layout generation

Each generated layout is a simulation scenario. To provide a comprehensive Pareto front, all potentially efficient layouts must be generated and evaluated by simulation. The number of aisles, cross-aisles, and cross-aisle types specify a layout. The model determines the lower and upper bounds of the numbers of aisles and cross-aisles and then enumerates all combinations of these two variables to generate the candidate layouts. For each candidate layout, two alternatives are considered: having all cross-aisles unidirectional or bidirectional. For each alternative, the sum of all travel distances within the layout is calculated, and the

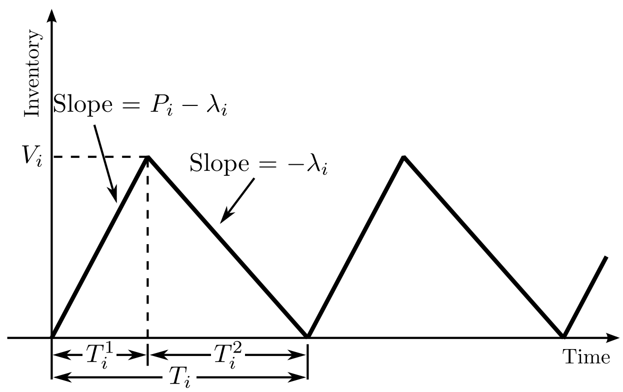

Figure 4.4: Changes in inventory of SKU i over its cycle time, $P_i > \lambda_i$.

layout with smaller total travel distance is selected for simulation. Note that both compared alternatives have the same storage area; therefore, the one with smaller total travel distance is a non-dominated solution among the two. Denote the lower and upper bounds of the numbers of aisles and cross-aisles by n_{min}, n_{max} ,c_{min}, and c_{max}, respectively. The total number of layouts evaluated by the simulation model is then $(c_{max} - c_{min} + 1)(n_{max} - n_{min} + 1)$.

Setting tight bounds on the numbers of aisles and cross-aisles significantly decreases the number of simulation scenarios and consequently decrease the computational time. For this reason, we develop a closed-form solution model in next section to obtain an optimal common bay depth that maximizes space utilization. This model is then used to define the bounds on the number of aisles.

4.4.1.1 Space Utilization

Consider SKU i that is produced in batches of Q_i pallets and stored at rate P_i pallets per unit of time. Assume it is retrieved from the storage lanes at rate λ_i pallets per unit of time, where $P_i > \lambda_i$ and replenishment starts when the inventory of the SKU reaches zero. Pallets of this SKU are H_i feet high and can be stacked up to Z_i pallets. The changes in the inventory of this SKU is shown in Figure 4.4. As it was shown in [7], the average waste of storage volume incurred to the warehouse by this SKU is given by

$$\bar{W} = AS^hS^wn + \frac{S^h}{2}\sum_{i\in I} x_i$$
$$+ \sum_{i\in I}\left(\frac{1}{2P_iZ_i}\right)\big((Q_i(S^h - Z_iH_i) - Z_iH_i)(P_i - \lambda_i) - \lambda_i(2S^h - Z_iH_i)\big), \quad (4.1)$$

where S^w is the warehouse width (in units of pallets), S^h is the warehouse clear height (in units of distance, i.e., inch, feet, etc.), n is the number of aisles, A is the aisle width (in units of pallets), x_i is assigned lane depth to SKU i, and I is the set of all SKUs.

The set of optimal bay depths that minimizes the total waste of storage volume is obtained by optimizing (4.1) with respect to x_i and n (note that these two variables are dependent, i.e. the sum of aisle widths and bay depths adds up to the warehouse length). Therefore, the constant part of (4.1) can be ignored in the optimization process. To develop a closed-form solution for the optimal bay depth, consider a common bay depth denoted by $\bar{x}$. It follows that

$$\sum_{i\in I} x_i = N_s\bar{x}, \quad (4.2)$$

where N_s is the number of SKUs stored in the warehouse. Thus, the optimal common bay depth is obtained by solving the following one-constraint optimization problem:

$$\text{Minimize} \quad AS^hS^wn + \left(\frac{1}{2}\right)S^hN_s\bar{x}. \quad (4.3)$$

Subject to:

$$2n\bar{x} + nA = S^l, \quad (4.4)$$

where S^l is the length of the warehouse (in units of pallets). Constraint (4.4) guarantees that the sum of bay depths and aisle widths are equal to the warehouse length. Solving (4.4) for $\bar{x}$ and substituting $\bar{x}$ in the objective function (4.3) gives an unconstrained optimization model whose objective function is

$$\text{Minimize} \quad AS^h S^w n + \left(\frac{1}{4n}\right) S^h (S^l - nA). \tag{4.5}$$

Differentiating (4.5) with respect to n, setting the results equal to zero, and solving for n gives the optimal number of aisles:

$$n^* = \sqrt{\frac{S^l N_s}{4 S^w A}}. \tag{4.6}$$

Consequently the optimal common bay depth is

$$\bar{x}^* = \sqrt{\frac{S^l S^w A}{N_s}} - \frac{A}{2}. \tag{4.7}$$

Expression (4.5) is continuously differentiable and has one extreme point. Hence, it is a unimodal function. The optimal number of bays must be integer. So, to obtain integer n^*, the two nearest integers smaller and greater than n^* are evaluated in (4.5) and the one that obtains the smaller value is chosen. Once n^* is determined, $\bar{x}^*$ is obtained by

$$\bar{x}^* = \frac{S^l - n^* A}{2n^*}. \tag{4.8}$$

If $\bar{x}^*$ is not integer, then it is not possible to divide the warehouse layout into exactly $2n^*$ equally deep bays. In this case, the initial bay depths are set to $\lfloor \bar{x}^* \rfloor$ for all bays. Then, the remaining $S^l - n^*(2\lfloor \bar{x}^* \rfloor + A)$ pallet positions are split into the bays one by one starting from the first bay.

From the operational point of view, deep lanes decrease availability of the storage space as it takes longer to fully deplete or replenish a deep lane. This issue is magnified especially when many SKUs are stored in the warehouse. In such situation, the goal is to keep lanes available (empty) as much as possible to provide storage space to incoming SKUs. The following proposition considers designing a layout from this perspective.

Proposition 4.1. *n^* maximizes the space availability in the layout.*

Proof. Once a lane is fully occupied, $S^h x_i$ storage volume becomes unavailable for the period that the lane is partially or fully occupied. As described in [7], the total lane-time that SKU i occupies in T_i^1 is

$$\left(\frac{1}{P_i - \lambda_i}\right)\left((I_i^{max} - 1) + (I_i^{max} - Z_i x_i - 1) + (I_i^{max} - 2Z_i x_i - 1) + \cdots + (I_i^{max} - K_i Z_i x_i)\right), \tag{4.9}$$

where I_i^{max} is the maximum inventory of SKU i and obtained by

$$I_i^{max} \approx \frac{Q_i\left(P_i - \lambda_i\right)}{P_i}, \tag{4.10}$$

and K_i is the number of required lanes for storage and is given by

$$K_i \approx \frac{Q_i\left(P_i - \lambda_i\right)}{P_i Z_i x_i}. \tag{4.11}$$

Similarly, the total lane-time that SKU i occupies in T_i^2 is calculated by

$$\left(\frac{1}{\lambda_i}\right)\left(I_i^{max} + (I_i^{max} - Z_i x_i) + (I_i^{max} - 2Z_i x_i) + \cdots + (I_i^{max} - K_i Z_i x_i)\right). \tag{4.12}$$

Adding (4.9) and (4.12) gives the total lane-time that SKU i occupies in its cycle time. Multiplying the result by $S^h x_i$ gives the total occupied volume-time and multiplying the result by λ_i/Q_i, the cycle time of SKU i, gives the average unavailable storage space that is occupied by SKU i

$$U_i = \frac{S^h x_i}{2} + \frac{S^h(Q_i(P_i - \lambda_i) - 2\lambda_i)}{2P_i Z_i}. \tag{4.13}$$

Adding unavailable space due to the aisle accessibility to the variable part of (4.13) gives the total unavailable space in the warehouse

$$\bar{U} = AS^h S^w n + \left(\frac{S^h}{2}\right) \sum_{i \in I} x_i. \tag{4.14}$$

Expression (4.14) is equivalent to the variable part of (4.1). Assuming a common bay depth, it converts to (4.3) and n^* that maximizes the space availability is obtained. □

4.4.1.2 Determining the bounds on the numbers of aisles and cross-aisles

Using the optimal common bay depth model (4.6), we set $n_{min} = \alpha n^*$ and $n_{max} = \beta n^*$. Denote the minimum and maximum number of lanes between the two subsequent cross-aisles by L_{min} and L_{max}, respectively. Then, $c_{min} = (S^w + L_{max})/(L_{max} + 2C)$ and $c_{max} = (S^w + L_{min})/(L_{min} + 2C)$, where C is the width of a unidirectional cross-aisle in units of pallets. Setting narrow bounds decreases the number of simulation scenarios at the risk of removing potential solutions.

We developed a numerical experiment to define efficient ranges for the allowable numbers of aisles and cross-aisles and to find the critical values for α, β, L_{min}, and L_{max}. We tested six test problems ranging from 10 to 300 SKUs using wide ranges for the allowable umber of aisles and cross-aisles. We set $\alpha = 0.8$, $\beta = 1.5$, $L_{min} = 10$, and $L_{max} = 40$, and run the algorithm by following the steps presented in Algorithm 4.1. The parameters of the algorithm and main properties of the Pareto fronts are displayed in Table 4.1 for all test problems.

Parameters n^o_{min} and n^o_{max} in Table 4.1 are the smallest and largest number of aisles observed in solutions of a Pareto front, respectively; c^o_{min} and c^o_{max} are the smallest and largest number of cross-aisles observed in solutions of a Pareto front, respectively. As Table 4.1 shows, $n^o_{min} = n^*$ for all test problems. This is because decreasing the number of

Table 4.1: The parameters of the algorithm and properties of the Pareto fronts for preliminary experiments.

Test problem	# of aisles					# of cross-aisles			
	n^*	n_{min}	n_{max}	n^o_{min}	n^o_{max}	c_{min}	c_{max}	c^o_{min}	c^o_{max}
10 SKUs	2	1	3	2	3	2	5	2	5
50 SKUs	3	2	5	3	5	2	10	2	9
100 SKUs	5	4	8	5	8	2	12	2	5
150 SKUs	6	4	9	6	9	2	15	2	9
200 SKUs	7	5	11	7	11	2	17	2	5
300 SKUs	8	6	12	8	12	2	20	2	7

aisles beyond n^* deteriorates both objective functions. Therefore, we set $\alpha = 1$ in our final experiment.

Increasing the number of aisles decreases utilization of the storage space but improves the total travel distance. However, as highlighted in [6], this improvement declines as the number of aisles grows and becomes insignificant once the layout has many aisles. Beyond this point, increasing the number of aisles does not justify loss of space. Taking this into account, we set $\beta = 1.4$.

The c^o_{max}s are smaller than c_{max}s in almost all test problems except 10 SKUs test problem for which it is equal to c_{max}. This shows that c_{max} can be efficiently shrunk without taking the risk of removing potential good solutions. The largest L_{min} that yields a c_{max} equal or larger than c^o_{max} for all test problems is obtained by setting $L_{min} = (0.1)S^w$. This set-up limits c_{max} while preserving all solutions in Pareto fronts of all test problems (this includes 10 SKUs problem as well for which it yields $c_{max} = 5$, i.e. no scenario reduction is achieved for this test problem). We set $c_{min} = 2$ because we assumed one cross-aisle exists next to the long sides of the warehouse (see assumption 9).

4.4.2 Transportation costs

For the sake of computational efficiency, we pre-calculate the rectilinear shortest distances between all potential origin and destinations pair in the layout and provide them to the simulation model as data matrices. Figure 4.5 presents the relative locations and

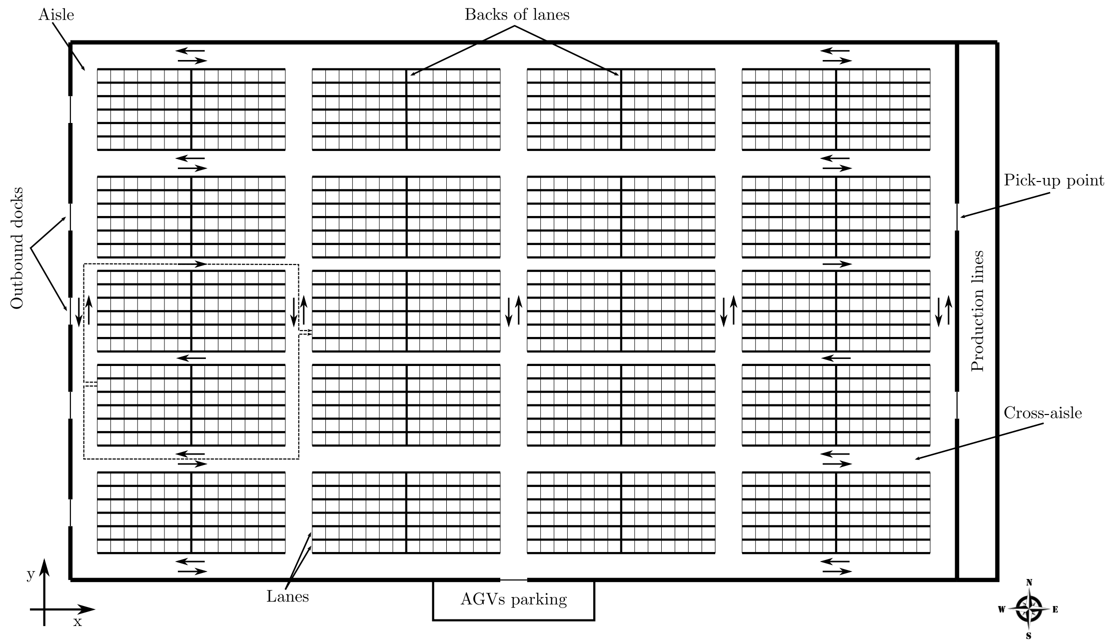

Figure 4.5: Components of the layout designed in this paper and their configurations.

directions of the important components in a layout. Note that the locations of production lines, vehicle parking, and outbound docks, as well as the number of P/D points, are given to the model and they can be different from those presented in Figure 4.5. Although the main purpose of the model is not to optimize the location of these components, one can analyze their arrangement by simulating different scenarios with different locations for these components while keeping the remaining design factors fixed. The following distances are calculated:

- distances between all storage lanes.
- distances from storage lanes to the production line and vice versa.
- distances from storage lanes to outbound docks and vice versa.
- distances from storage lanes to the vehicle parking and vice versa.
- distances from outbound docks to the production line.
- distances from outbound docks to the vehicle parking.

- distances from the vehicle parking to the production line.

A rectilinear distance between two locations is obtained by adding the distances traveled along the x-axis and y-axis. As Figure 4.5 illustrates, we used a Cartesian coordinate system and assumed the origin is located at the southwest corner of the layout. Instead of designing a network of locations and finding the shortest path, we take advantage of the special shape of the layout to simplify the distance calculations as explained in the following.

The proposed model considers both unidirectional and bidirectional cross-aisles. This adds an extra level of complexity to calculating the distances. When both origin and destination are surrounded by the same two cross-aisles, the shortest path is selected from the two paths that connect them from the closest cross-aisles located at the north or south of the origin (see Figure 4.5). Therefore the distance to the closest cross-aisles located at the north or south of all origin points must be taken into account. Also, the travel direction (west to east or vice versa in Figure 4.5) must be considered because the closest cross-aisle may be a unidirectional cross-aisle heading to the opposite direction. Note that the shortest path is the path that travels through at most two aisles (aisles that origin and destination points are located in) and one cross-aisle. For the sake of computational efficiency, the other paths are not evaluated.

Consider a west-to-east trip. The shortest path through the cross-aisle located at the north of the origin is obtained by

$$D^N_{W\to E} = \begin{cases} |y^c_i - y^o| + \left|y^c_i - y^d\right| + x^d - x^o & \text{if } i \notin \emptyset, \\ \infty & \text{otherwise,} \end{cases} \tag{4.15}$$

where i is the index of the closest cross-aisle located at the north of the origin heading east, y^c_i is the y-coordinate of the cross-aisle i, y^o is the y-coordinate of the origin, y^d is the y-coordinate of the destination, and x^o and x^d are the x-coordinates of the origin and destination points, respectively. If such a cross-aisle does not exist (i.e. there is no cross-aisle at the north of the origin or they are all unidirectional heading opposite directions) then the distance is set to infinity.

Similarly, the shortest path from the southern cross-aisle is obtained by

$$D^S_{W\to E} = \begin{cases} \left|y^c_j - y^o\right| + \left|y^c_i - y^d\right| + x^d - x^o & \text{if } j \notin \emptyset, \\ \infty & \text{otherwise,} \end{cases} \tag{4.16}$$

where j is the index of the closest cross-aisle located at the south of the origin heading east. Consequently, the shortest path between these two locations is obtained by

$$D_{W\to E} = min\{D^N_{W\to E}\ ,\ D^S_{W\to E}\}. \tag{4.17}$$

Note that $D_{E\to W}$ is not necessarily equal to $D_{W\to E}$ because different paths may have to be taken to travel from east to west due to existence of unidirectional cross-aisles. Expressions (4.15)-(4.17) are valid for all distances between storage lanes. They are also valid for distances between storage lanes and pick-up points, outbound docks, and vehicle parking if they are located on the short sides of the warehouse. They are also valid for the distances between any pair of outbound docks, pick-up points, and vehicle parking if they are located on the short sides of the layout. If either of these components located on the long sides of the warehouse, the distance between the storage lane and that component is simply obtained by

$$D_{S\to N} = D_{N\to S} = \left|y^o - y^d\right| + \left|x^o - x^d\right|. \tag{4.18}$$

This is because aisles are bidirectional and the cross-aisles located in front of those locations are assumed to be bidirectional (see assumptions 6 and 9). Expression 4.18 is also used to calculate distances between pairs of outbound docks, pick-up points, and vehicle parking if either origin or destination is located on the long sides of the layout.

4.4.3 Simulation model

The core of the proposed model is an event-oriented simulation model that simulates pallet storage and retrieval operations in a warehouse while computing the performance metrics pertinent to the space utilization and transportation costs. The model consists of nine procedures: three events to simulate a replenishment operation, three events for a retrieval operation, two events for a vehicle release, and a warm-up event.

In a replenishment operation, a vehicle picks up a produced (or inbound arrival) pallet from the production line (or inbound dock) and delivers it to a storage lane. The retrieval operation is referred to an operation in which a vehicle picks up a pallet from the storage area and delivers it to an outbound dock. The simulation events are:

- Production pick-up: The closest available vehicle to the pick-up point (production line or inbound dock) is dispatched to pick up a waiting pallet, and the "lane drop-off" event is scheduled taking the travel distance into account.

- Lane drop-off: The dispatched vehicle picks up the pallet from the production line or inbound dock and starts traveling to the assigned storage lane. The "replenishment" event is scheduled taking the travel distance into account.

- Replenishment: The pallet is stored in the target lane, and the "release vehicle" event is scheduled at the simulation time plus epsilon time unit.

- Outbound pick-up: The closest vehicle to the pick-up lane is dispatched to pick up the requested SKU from its assigned lane. The "retrieval" event is scheduled taking the travel distance into consideration.

- Retrieval: The dispatched vehicle picks up the requested pallet from the floor stack and starts traveling to the assigned outbound dock. The "truck drop-off" event is scheduled considering the travel time.

- Truck drop-off: The requested SKU is delivered to the assigned outbound dock and "release vehicle" event is scheduled at the simulation time plus epsilon time unit.

- Release vehicle: The empty vehicle starts traveling to the parking. The "Park vehicle" event is scheduled considering the travel distance.

- Park vehicle: The dispatched vehicle is parked and becomes available.

- Warm-up: This event is executed once and resets all variables used for performance evaluation (not the control variables) to their initial values.

Note that the "Release vehicle" event is scheduled at the simulation time plus epsilon time unit to allow waiting requests for pick-up to be processed earlier (continuous pick-up operations). Interested readers are referred to [6] for more details on the simulation model.

The simulation model particularly computes two performance metrics: required number of labors/vehicles in the warehouse (C_l), and percentage of wasted space (C_s). The required number of labors/vehicles is obtained by summing the distance traveled by all vehicles in the simulation and then dividing the result by the total distance that a labors/vehicles can travel in the simulated period. That is,

$$C_l = \frac{d_u + d_l}{V * (T^s - T^w)}, \tag{4.19}$$

where d_u, and d_l are the total loaded and unloaded distances traveled by all vehicles in simulation, T^s is the simulation time, T^w is the warm-up time, and V is the average speed of a vehicle/labor in the warehouse. The percentage of wasted space is obtained by summing the total wasted space in lanes during simulation and dividing it by the total space-time of the warehouse. That is,

$$C_s = \frac{S^h(nAS^w + cCS^l)(T^s - T^w) + \sum_{i=1}^{N_b} \sum_{j=1}^{N_l} W_{ij}^H}{(S^w S^l S^h)(T^s - T^w)}, \tag{4.20}$$

Table 4.2: Computational experiments.

Problem	Warehouse size (ft)	n_{min}	n_{max}	c_{min}	c_{max}	# of simulated layouts	# of layouts in Pareto front	Avg. Simulation time per scenario (sec)	Computational time (sec)
10 SKUs	200 × 400	2	3	2	8	14	5	16	67
50 SKUs	400 × 720	3	5	2	10	27	9	21	65
100 SKUs	520 × 960	5	7	2	10	27	8	108	330
200 SKUs	760 × 1400	7	10	2	10	36	12	110	448
300 SKUs	920 × 1600	8	12	2	11	50	18	412	2117
400 SKUs	1080 × 1840	9	13	2	11	50	12	257	1321
500 SKUs	1200 × 2080	10	14	2	11	50	16	340	1780
600 SKUs	1320 × 2200	11	16	2	11	60	19	1070	6665
700 SKUs	1440 × 2400	12	17	2	11	60	24	547	3458
800 SKUs	1600 × 2600	13	19	2	11	70	24	548	18505
900 SKUs	1720 × 2880	14	20	2	11	70	25	776	11580
1000 SKUs	1840 × 3000	14	20	2	11	70	24	1148	17403

where c is the number of cross-aisles in the layout, C is the cross-aisle width, N_b is the number of bays in the layout, N_l is the number of lanes in a bay, and W_{ij}^H is the honeycombing waste at jth lane of bay i during simulation. All simulation scenarios are compared with respect to these two factors and non-dominated solutions form the Pareto front.

4.5 Experimental analysis

In this section, we design an experimental study to analyze efficiency of our model in terms of computational time and also comprehensibility of the generated Pareto fronts. We tested the algorithm on 12 randomly generated test problems introduced in [7]. The size of the selected test problems vary from small (10 SKUs) to industrial-sized (1000 SKUs) to better study the effect of the warehouse size on the computational time.

The proposed model was coded with Python 2.7.11 and run on the Auburn University Hopper Cluster on Intel Xeon processors E5-2660 (2.6GHz) with 128 GB of RAM. We ran all scenarios in parallel on ten cores. We set warm-up period to one month, start-up inventories to zero, and simulation time to 8 months as described in [6]. We used common random numbers among scenarios for variance reduction.

Table 4.2 shows the computational times and the numbers of simulated scenarios for all test problems. The simulation module itself took on average 16 seconds for the smallest problem to 1148 seconds for the largest problem to evaluate a layout. The total computational

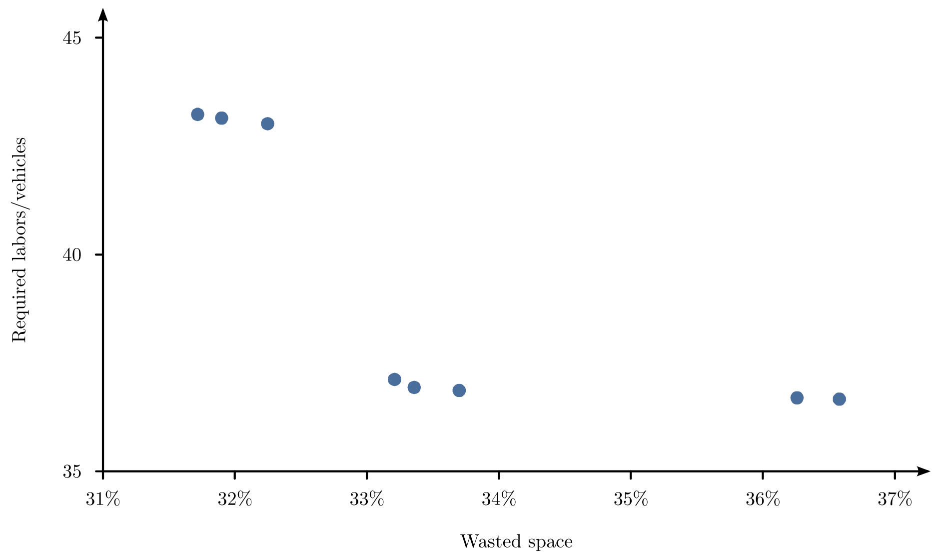

Figure 4.6: Pareto front of the 100 SKUs test problem.

time grows from 67 seconds for the smallest problem to 17403 seconds for the largest test problem. One should note that since the simulation scenarios were run in parallel, the total computational time is much smaller than the sum of simulation time for all scenarios. From the computational perspective, it can be seen that despite the fact that multiple scenarios are simulated for a steady state; the algorithm is capable of finding the optimal Pareto front in a reasonable time for large size test problems.

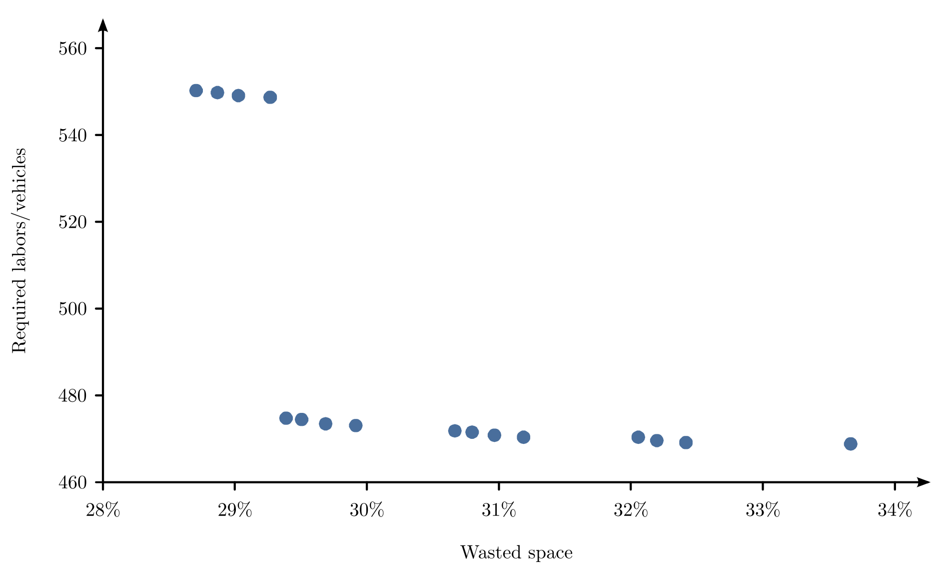

Figure 4.7: Pareto front of the 500 SKUs test problem.

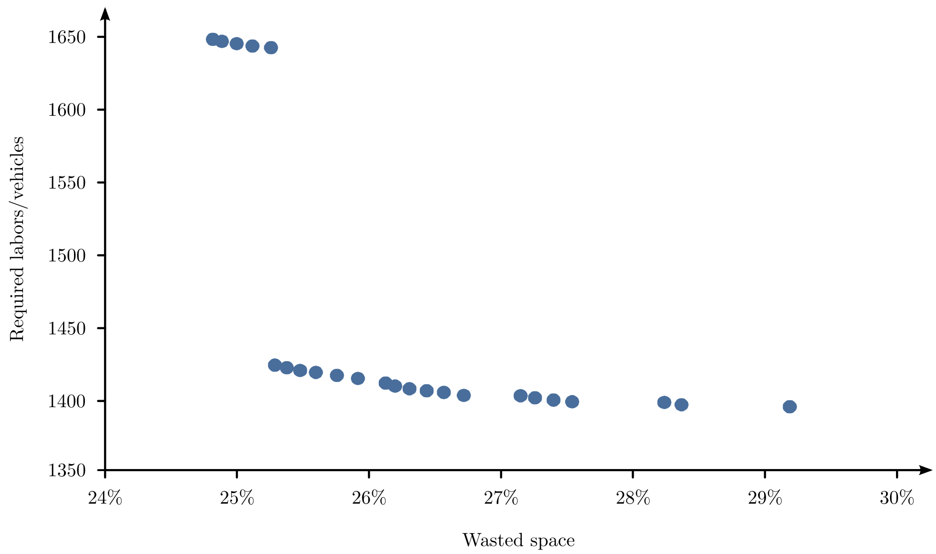

Figure 4.8: Pareto front of the 1000 SKUs test problem.

Table 4.3: Solutions of the Pareto fronts for 100, 500, and 1000 SKUs test problems.

Layout	100 SKUs			500 SKUs			1000 SKUs		
	# aisles	# cross.	cross. type	# aisles	# cross.	cross. type	# aisles	# cross.	cross. type
1	4	4	Uni	10	4	Uni	14	4	Uni
2	5	4	Uni	11	4	Uni	15	4	Uni
3	6	4	Uni	12	4	Uni	16	4	Uni
4	4	3	Bi	13	4	Uni	17	4	Uni
5	5	3	Bi	9	16	Uni	18	4	Uni
6	6	3	Bi	11	3	Bi	14	3	Bi
7	5	5	Bi	12	3	Bi	15	3	Bi
8	6	5	Bi	13	3	Bi	16	3	Bi
9	—	—	—	10	5	Bi	17	3	Bi
10	—	—	—	11	5	Bi	18	3	Bi
11	—	—	—	12	5	Bi	19	3	Bi
12	—	—	—	13	5	Bi	14	5	Bi
13	—	—	—	11	7	Bi	15	5	Bi
14	—	—	—	12	7	Bi	16	5	Bi
15	—	—	—	13	7	Bi	17	5	Bi
16	—	—	—	13	9	Bi	18	5	Bi
17	—	—	—	—	—	—	19	5	Bi
18	—	—	—	—	—	—	16	7	Bi
19	—	—	—	—	—	—	17	7	Bi
20	—	—	—	—	—	—	18	7	Bi
21	—	—	—	—	—	—	19	7	Bi
22	—	—	—	—	—	—	18	9	Bi
23	—	—	—	—	—	—	19	9	Bi
24	—	—	—	—	—	—	19	11	Bi

We selected a small (100 SKUs), medium (500 SKUs) and large (1000 SKUs) test problem to analyze their respective Pareto fronts. Figures 4.6, 4.7, and 4.8 demonstrate the Pareto fronts for these test problems, respectively. Table 4.3 shows solutions in these Pareto fronts in order of their appearances from left to right. The following observations are highlighted.

- Solutions in the Pareto fronts can be clustered into two groups. The first group covers few solutions at the top left of the Pareto front. These solutions mainly incur less wasted space to the warehouse at the cost of higher transportation costs. The second group consists of larger portion of solutions and covers the ones located at the bottom of the Pareto front. These solutions mainly incur smaller transportation costs but higher waste of space. The first group will be optimal when the space unit cost (rent, maintenance,...) is considerably higher than the transportation unit cost (labors/vehicles, maintenance ,...). If both costs are almost equal or the transportation unit cost is higher, then the second group of solutions are optimal. Adding a new cross-aisle or aisle decreases the transportation costs a little among the solutions in this group while the utilization of storage space exacerbates considerably.

- Bidirectional cross-aisles are more efficient than the unidirectional cross-aisles when the warehouse has more than three cross-aisles. Note that the traffic congestion has not been taken into account.

- Increasing the number of cross-aisles from two to three makes significant improvement in transportation costs while wasted space deteriorates not as much. Adding more cross-aisles improves the transportation costs, but the impact becomes less significant as more cross-aisles are added whereas the utilization of the storage space keeps decreasing considerably. Therefore, adding many cross-aisles does not necessarily improve transportation costs as pickers have to traverse the cross-aisles themselves too.

- Fixing the number of cross-aisles, transportation costs (or travel distance) decrease as the number of aisles increases (consequently bay depths decrease in the layout); However, utilization of the storage space decreases as the number of aisles increases. This is in line with the findings of [6].

- Although increasing the number of aisles improves the transportation costs, the improvement is not significant comparing to adding a new cross-aisle.

- The number of solutions in a Pareto front increases as the size of the warehouse grows. This is because the number of allowable set-ups for the layout increases.

4.6 Conclusions

In this paper, we developed a simulation-based optimization algorithm to simultaneously optimize utilization of the storage space and transportation costs through designing a layout for block stacking warehouses. We developed a closed-form solution to find a common bay depth to maximize utilization of the storage space in the warehouse. Our approach finds the optimal number of aisles, cross-aisles and cross-aisle type for a block stacking warehouse. The model provides the Pareto front to the decision maker.

Our exhaustive computational experiment shows that the model finds the Pareto front in a reasonable time for large-sized test problems. Analyzing Pareto fronts shows that although adding new cross-aisles improves the total travel distance in a warehouse, the improvement rate decreases as more cross-aisles are added to the layout whereas reduction in utilization of storage space continues. Hence adding new cross-aisles beyond some level does not justify loss of storage space even if the transportation unit cost is higher than the space unit cost.

The solutions in Pareto fronts can be clustered into two clusters: layouts that are highly utilized but less efficient in terms of transportation and vice versa (no solution at the middle). Layouts in the first group contain two to three cross-aisles whereas the solutions in the next group have many cross-aisles up to ten. Our experiments also show that bidirectional cross-aisles are in general more efficient in terms of the travel distances especially when more than three cross-aisles exist in the layout.

The number of aisles and consequently bay depths affect the transportation costs in the warehouse in addition to the utilization of the storage space. Increasing the number of aisles in a layout leads to reduction in the travel distance; However, this reduction becomes less significant as the number of aisles grows.

Bibliography

[1] P. Baker and M. Canessa. Warehouse design: A structured approach. *European Journal of Operational Research*, 193(2):425–436, 2009.

[2] J. J. Bartholdi and S. T. Hackman. *Warehouse & Distribution Science: Release 0.89*. Supply Chain and Logistics Institute, 2008.

[3] L. F. Cardona, D. F. Soto, L. Rivera, and H. J. Martínez. Detailed design of fishbone warehouse layouts with vertical travel. *International Journal of Production Economics*, 170, Part C:825–837, 2015.

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

[5] S. Derhami, J. S. Smith, and K. R. Gue. Optimising space utilisation in block stacking warehouses. *International Journal of Production Research*, pages 1–17. doi: 10.1080/00207543.2016.1154216.

[6] S. Derhami, J. S. Smith, and K. R. Gue. A simulation model to evaluate the layout for block stacking warehouses. *Progress in Material Handling Research: 2016*, 2016.

[7] S. Derhami, J. S. Smith, and K. R. Gue. Space-efficient layouts for block stacking warehouses. *Manuscript Submitted to European Journal of Operational Research*, 2017.

[8] M. Goetschalckx and D. H. Ratliff. Optimal lane depths for single and multiple products in block stacking storage systems. *IIE Transactions*, 23(3):245–258, 1991.

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

[10] J. Gu, M. Goetschalckx, and L. F. McGinnis. Research on warehouse design and performance evaluation: A comprehensive review. *European Journal of Operational Research*, 203(3):539–549, 2010.

[11] K. R. Gue and R. D. Meller. Aisle configurations for unit-load warehouses. *IIE Transactions*, 41(3):171–182, 2009.

[12] K. R. Gue, G. Ivanović, and R. D. Meller. 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, 2012.

[13] X. Guo, Y. Yu, and R. B. de Koster. Impact of required storage space on storage policy performance in a unit-load warehouse. *International Journal of Production Research*, 54(8):2405–2418, 2016.

[14] D. Kind. Elements of space utilization. *Transportation and Distribution Management*, 15:29–34, 1975.

[15] T. Lamballais, D. Roy, and M. de Koster. Estimating performance in a robotic mobile fulfillment system. *European Journal of Operational Research*, 256(3):976–990, 2017.

[16] T. N. Larson, H. March, and A. Kusiak. A heuristic approach to warehouse layout with class-based storage. *IIE Transactions*, 29(4):337–348, 1997.

[17] J. Li, M. Moghaddam, and S. Y. Nof. Dynamic storage assignment with product affinity and ABC classification—a case study. *The International Journal of Advanced Manufacturing Technology*, 84(9):2179–2194, 2016.

[18] J. O. Matson. *The analysis of selected unit load storage systems*. PhD thesis, Georgia Institute of Technology, Atlanta, GA, 1982.

[19] M. Moshref-Javadi and M. R. Lehto. Material handling improvement in warehouses by parts clustering. *International Journal of Production Research*, 54(14):4256–4271, 2016.

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

[21] O. Öztürkoğlu, K. R. Gue, and R. D. Meller. Optimal unit-load warehouse designs for single-command operations. *IIE Transactions*, 44(6):459–475, 2012.

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

[23] J. A. Pazour and R. D. Meller. An analytical model for A-frame system design. *IIE Transactions*, 43(10):739–752, 2011.

[24] F. Ramtin and J. A. Pazour. Analytical models for an automated storage and retrieval system with multiple in-the-aisle pick positions. *IIE Transactions*, 46(9):968–986, 2014.

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

[26] K. J. Roodbergen and R. de Koster. Routing methods for warehouses with multiple cross aisles. *International Journal of Production Research*, 39(9):1865–1883, 2001.

[27] K. J. Roodbergen and I. F. A. Vis. A model for warehouse layout. *IIE Transactions*, 38(10):799–811, 2006.

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

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

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

[31] L. M. Thomas and R. D. Meller. Analytical models for warehouse configuration. *IIE Transactions*, 46(9):928–947, 2014.

[32] L. M. Thomas and R. D. Meller. Developing design guidelines for a case-picking warehouse. *International Journal of Production Economics*, 170, Part C:741–762, 2015.

[33] T. S. Vaughan and C. G. Petersen. The effect of warehouse cross aisles on order picking efficiency. *International Journal of Production Research*, 37(4):881–897, 1999.

[34] S. Zhou, Y. Y. Gong, and R. de Koster. Designing self-storage warehouses with customer choice. *International Journal of Production Research*, 54(10):3080–3104, 2016.

Chapter 5

Concluding remarks

This dissertation studied designing optimal layouts for block stacking warehouses from space utilization and transportation cost perspectives. We developed models to find optimal lane depths for block stacking in diverse environments and proposed multiple optimization models to design space and transportation-efficient layouts for this storage system. We developed a diverse set of analytical and statistical approaches, including non-constraint optimization, mixed integer programming, and event-based simulation, to address the literature gaps in designing block stacking warehouses.

We proposed three closed-form solution models to minimize wasted space in diverse manufacturing and non-manufacturing environments by finding the optimal lane depth. We showed that although our proposed are built assuming deterministic production and demand rates, the resulting lane depths are robust and remain near optimal under uncertainty. We demonstrated that using an infinite production rate model in a finite production rate system results in lane depths about twice as deep as they should be. However, the resulting waste of volume is not significant because the space utilization curve, as a function of lane depth, is quite flat as the lane depth increases.

We proved that the optimal lane depth model underestimates the accessibility waste for the layout design problem and therefore, cannot be used to calculate optimal bay depths for a space-efficient layout. We developed two models to find the optimal bay depths for a space-efficient layout based on the problem size. We showed that the random SKU assignment policy is not an optimal policy when the layout has multiple bay depths. We found that accessibility waste outweigh honeycombing waste in their trade-off and a layout with deep bays better utilizes the storage space.

Our simulation results highlighted that both space utilization and transportation costs must be taken into account to find the optimal number of cross-aisles for a layout. Motivated by this finding, we developed a multi-objective simulation-based optimization algorithm to find the optimal number of cross-aisles and aisles in a layout. Our experimental study showed that the improvement rate in transportation cost decreases as more cross-aisles are added to the layout, and that the solutions of a Pareto front for this multi-objective problem are clustered into two groups: layouts that are highly space utilized but less efficient in terms of transportation; and, vice versa (there is no solution in the middle). We found that bidirectional cross-aisles are in general more efficient in terms of the total travel distance especially when more than three cross-aisles exist in the layout. We also showed that increasing the number of aisles in a layout results in a reduction in total travel distance; however, this reduction becomes less significant as the number of aisles grows.

This dissertation aimed to provide comprehensive and accurate models to address the layout design problem for block stacking warehouses from three different perspectives motivated from real-world situations. While we relaxed many restrictive and unrealistic assumptions of the previous studies and covered the major gaps in the literature, there exist promising extensions to our models that would further enhance performance of a warehouse layout. These extensions are:

- **Flexible layout:** We assumed all lanes in a bay have the same depth. While this assumption reduces the computational difficulty of the layout design problem and facilitates transportation within the warehouse, relaxing it provides more depth choices to SKUs and would improve utilization of the storage space.

- **Traffic congestion:** We indirectly modeled traffic congestion by adding stochastic variations to the travel times. Although this simplification does not undermine the main findings of this dissertation, modeling the traffic congestion may influence the warehouse throughput. Considering the traffic congestion in determining the number and type of cross-aisles would further enhance the layout performance in real-world situations.

- **Robust design:** Designing a robust layout that remains optimal over changes in influential factors like the demand and production over multiple periods is another future direction that benefits systems with high level of uncertainty.

We believe the models presented in this dissertation along with the future research directions described above will help companies to design their warehouses more efficiently and provide the researchers a deep insight to investigate further the design of block stacking warehouses.