MAXIMIZING THE TRAVEL TIME EFFICIENCY OF THE AUTOMATED STORAGE AND RETRIEVAL SYSTEM OF COMPANY X

Daan Veuger

August, 2024

Master Thesis
Industrial Engineering & Management

UNIVERSITY OF TWENTE.

Master Thesis
Industrial Engineering & Management

Production and Logistics Management – *Supply Chain and Transportation Management*
Financial Engineering and Management – *Valuation and Risk Management*

Author
Daan Veuger
University of Twente

Company
Company X
Confidential

Supervisors
University of Twente

Dr. L. Xie (Lin)
Faculty of Behavioural, Management and Social Sciences (BMS)
Industrial Engineering & Business Information Systems (IEBIS)

Dr. B. Alves Beirigo (Breno)
Faculty of Behavioural, Management and Social Sciences (BMS)
Industrial Engineering & Business Information Systems (IEBIS)

Company X

Confidential

UNIVERSITY
OF TWENTE.

Preface

Dear reader,

As I reach the end of my academic journey at the University of Twente, spanning five enriching years, three years as a bachelor’s student and the last two as a master’s student, I reflect on the unforgettable growth and development I have experienced both personally and professionally. Throughout this time, I have had the privilege of meeting and learning from many remarkable individuals. I am proud to complete this significant chapter in my life and am eager to embrace the opportunities that lie ahead. I look forward to furthering my personal and professional growth in my career as an industrial engineer.

This thesis marks the culmination of my time at the University of Twente. The research was conducted at Company X, where I focused on maximizing the efficiency of the Automated Storage and Retrieval System (AS/RS) by exploring various storage strategies. I am deeply grateful to Company X for providing me with this opportunity and for their trust and support throughout the research process. Though the journey was challenging, my dad used to say: “without friction no shine”.

I would like to extend my thanks to two individuals at Company X who we instrumental during my research. My first company supervisor, your faith and assistance were invaluable, offering guidance with ideas, data, and organizational solutions. My second company supervisor, our almost weekly meetings were crucial for discussing and resolving various issues. Thank you both for your support and encouragement.

Additionally, I wish to express my gratitude to dr. Lin Xie for her supervision during the past semester. Your availability and insightful answers to my questions were greatly appreciated. I also want to thank dr. Breno Alves Beirigo, my second supervisor, for his important role despite our limited interactions. Your contributions were no less significant. Now all that is left is to wish you an enjoyable time during reading this research!

Daan Veuger

Raalte, July 2024

Management Summary

Company X has been experiencing rapid growth in the production of nutritional healthcare products, a trend expected to continue in the coming years. To support this expansion, the company decided to construct a new raw materials warehouse, apart from the production location. The goal is to put off the pressure from the warehouse department at the production location itself. This new warehouse gets an advanced Automated Storage and Retrieval System (AS/RS). This multi-deep AS/RS offers high storage density, but presents challenges in accessing storage locations directly. The efficiency of this system is largely determined by the way how ingredients are stored, leading to the central research question:

"How do we design an optimal storage assignment for the automated storage/retrieval system of Company X such that the travel time efficiency is maximized?"

Solution

To address this question, we collected and analyzed production data to identify the most frequently accessed ingredients. An Linear Programming optimization model was initially developed with as objective to minimize the total travel time to handle all storage and retrieval requests. However, this proved insufficient for the complexity of the problem. Therefore, we created a simulation model that simulates the AS/RS and is able to determine the exact travel times when travelling from a certain location to any other location. This simulation model is able to handle the larger problem size, in which we mapped initial stock and processed storage and retrieval requests afterwards. We also implemented two metaheuristic methods to refine the solutions further.

Results

Two sets of problem instances were tested, which we call the small problem instance and the complete problem instance. The small problem instance was tested using both the optimization model and the simulation model, while the complete instance was tested exclusively with the simulation model. The purpose of analyzing both instances was to assess the performance of the complete problem instance by leveraging insights gained from the small instance through the mathematical model. The following characteristics define the small and complete problem instances.

Type of instance	Initial stock (pallets)	Number of products	Number of retrieval requests	Number of storage requests
Small	1,951	8	40	70
Complete	4,242	83	8,512	10,122

The results for the small problem instance are shown directly below. The total number of handled pallets equals 110, and all numbers are shown in minutes. The optimization method that yielded the best results was the Simulated Annealing algorithm, which improved the initial solution of the class-based storage strategy by 38.58%.

Storage Strategy (small instance)	Total travel time (minutes)	Average travel time (minutes)	Average travel time after optimization (minutes)
Exact	119.33	1.08	-
Random	295.25	2.68	1.60
Class-based	233.25	2.12	1.30
Dedicated	210.53	1.91	-

The next set of results are related to the complete problem instance. The total number of handled pallets in this case equals 18,634. The percentage improvement after optimization is much smaller for the complete instance compared to the small instance. A major reason for this is that retrieval on batch number is preferred, with as result that not always the ingredient stored on the closest location is retrieved.

Storage Strategy (complete instance)	**Total travel time (hours)**	**Average travel time (minutes)**	**Average travel time after optimization (minutes)**
Random	582.27	1.87	-
Class-based	504.46	1.62	1.58
Dedicated	534.63	1.72	--

Furthermore, an utilization analysis was performed to assess the capabilities of the AS/RS to store large number of pallets at the same time. In accordance with the stakeholders of Company X, it was decided that at most 1 type of ingredient can be stored in storage lanes that are accessible from a single side, and at most 2 types in storage lanes that are accessible by 2 sides. This ensures that all ingredients remain accessible at all times.

System utilization			
Stored based on	**Confidence level α**	**Lower bound**	**Upper bound**
Ingredient ID	95%	84.30%	84.74%
Lot number	95%	82.26%	82.85%

Another improvement method can be used to increase the utilization level further. We used a swap strategy that swaps certain storage lanes in order to create feasible storage locations. With this strategy, we are able to utilize the system up to 90.38%. An disadvantage of this is that this method must be performed on beforehand. Besides, a significant amount of swaps might be required to create extra storage space, which might not be possible within a short time period.

Recommendations
Based on the results and conclusions from the research, we are able to set up a list with recommendations that are shown below:

- With a high utilized system, the class-based storage strategy with 3 classes works best.
- With a low utilized system, the dedicated storage strategy works best, where the closest storage lanes are picked first for storage.
- Storing products based on their lot number does not decrease the maximum utilization rate significantly. Therefore, storage based on lot number is preferred over storage on ingredient ID, to prevent relocations.
- Investigate and verify the quality of the collected data.
- Reconsider the allergen storage constraints, since relaxing this constraint will enable the utilization rate to increase even further.

Table of Contents

List of tables

List of figures

Acronyms

Abbreviation	Meaning	First introduced on page
M	Million	1
XLOG	Company X Logistics	1
3PL	Third Party Logistics	1
AS/RS	Automated Storage/Retrieval System	IV
ROI	Return On Investment	2
SKU	Stock Keeping Unit	1
I/O	In/Out	6
Mm	Millimeter	7
GMP	Good Manufacturing Practice	9
FEFO	First Expired, First Out	10
ERP	Enterprise Resource Planning	10
MILP	Mixed Integer Linear Programming	16
COI	Cube-per-Order Index	21
EOQ	Economic Order Quantity	22
DOS	Duration-of-Stay	22
SI	Swarm Intelligence	16
ACO	Ant Colony Optimization	17
VRP	Vehicle Routing Problem	17
TSP	Traveling Salesman Problem	17
GA	Genetic Algorithm	18
KPI	Key Performance Indicator	26
AGV	Automated Guided Vehicle	28
CI	Confidence Interval	38
SLAP	Storage Location Assignment Problem	30
SA	Simulated Annealing	44
VNS	Variable Neighborhood Search	46
LASRSP	Location Assignment and Storage/Retrieval Scheduling Problem	56

1 Introduction

In this chapter we start with the introduction of Company X. Especially the size of the company and what it produces is discussed. Thereafter in Section 1.2, we explain why this research is conducted. In Section 1.3, we discuss the problem that is faced by Company X. In the next section, we argue what we intend to reach at the end of the research. This research objective is followed by the research questions in Section 1.4, while we end with the scope in Section 1.5.

1.1 Company introduction

Company X is a pharmaceutical company operating worldwide together with its subsidiaries. Company X, hereafter referred to as Company X, develops, manufactures, and sells health care products. Company X has more than 120,000 employees worldwide and reaches over 160 countries. In the Netherlands, Company X is based at 8 locations, of which Overijssel has two of them. This research is conducted at one of the locations in Overijssel, and this location mainly focuses on the production of nutritional products. This segment provides adult and pediatric nutritional products.

The Company X location where this research takes place produces 885 stock keeping units (SKU's) and these products reach 67 different countries. 573 employees in the production plant and 75 regional support employees are responsible for the production at Company X. The nutritional products segment of Company X can be divided into two subsegments, namely powder and liquid. In 2023, Company X produced for 5.7M (Million) Lbs of powder products, while 3.2M Liter of liquid products were produced. Examples of powder products and liquid products can be found in Appendix A – Products of Company X.

1.2 Research motivation

The focus of the research lies at a location nearby the production location, where a new raw materials warehouse is built. The engineering department is responsible for this, in combination with the warehouse department and the operational readiness department. These latter two are elements of the engineering department. The operational readiness department ensures that personnel is trained, equipment is on the right place on the right time, and raw materials are present in the correct amounts.

The reason for the built of this new warehouse is mainly caused by an increase in demand for the products that are produced at Company X. To facilitate this increase in demand, Company X is expanding its production capacity, by adding an extra production line to the plant. To create enough space for this new production line, a part of the storage capacity was used. This resulted in more production capacity, but less storage space. Expanding the production location in Overijssel is not an option, since this location is landlocked.

1.3 Problem statement

Initially, Company X stored their raw materials in-house at the production location in Overijssel. As the demand for their products increased, the demand for raw materials increased as well. This increase caused that not all raw materials can be stored at the production location anymore. In the first place, the solution to this problem was solved by Company X Logistics (XLOG) in Breda. This distribution center initially served as a finished product center only, but now stores raw materials as well. XLOG Breda was not able to store all the finished products and raw materials at the same time, which caused that a Third Party Logistics (3PL) provider in Overijssel also stored some of the raw materials. In addition, a special safety warehouse located in Barneveld stores hazardous products which may not be stored at regular warehouses. All in all, the raw materials are located at many different locations, which resulted in high transportation and storage costs. The decision was made to build a new warehouse, that provides a centralized raw material storage and the possibility to prepare batch orders for the production plant. In addition, the new warehouse also gives the opportunity to comply with the increasing demand in the future.

As said, this new warehouse is still under construction. The size of the warehouse is already determined, along with the layout of the inside. One of the elements Company X introduces within this warehouse is an automated storage/retrieval system, also called an AS/RS or AS/R system. This system will occupy approximately two-thirds of the maximum storage capacity of this warehouse. Because Company X does not have an AS/RS yet, it wants to gain insight in the working of such a system. The physical layout and equipment of the AS/R system is inflexible, which means that it is essential to design it right first time. Doing things right at the first time is an important characteristic for Company X. This is also one of the "Lean" principles that employees are encouraged to work in accordance with. Designing the AS/RS system right at once results in a more resilient warehouse, higher throughput times and thus a higher return on investment (ROI). The physical layout of the AS/R system is already determined, meaning that the length, width, height, and thus maximum capacity is fixed. Subsequently, things that can be optimized are the routing of the storage and retrieval shuttles, and the pallet placement positions. Shuttles are small robots that drive autonomously over a movement lane and that are able to store and retrieve pallets in the AS/R system. Since the routing of the shuttles is already taken care of by the supplier of the AS/R system, we will focus on the pallet placement positions. Optimizing the pallet placement is a complex and difficult task, which brings us at the core problem of this research:

"It is unknown how the ingredients must be stored in the AS/R system at the new warehouse of Company X"

1.4 Research objective

The objective of the research is to provide Company X with a detailed advise about where to place which raw material within their AS/R system. This is achieved by using a simulation model in which the AS/R system is programmed according to parameters and constraints communicated by stakeholders of Company X. The positive effects that should be experienced by the management team of Company X are lower waiting time for shuttles and a higher overall productivity. Because the products are stored in an efficient way, the shuttles need less time on average to store or receive an order. Personnel that operate the inbound and outbound process of the AS/R system will be higher utilized. Another advantage of a lower storage and retrieval time is visible in the performance numbers during peak hours. By lowering the average storage and retrieval time, we increase the number of storage and retrieval requests per hour. If the storage assignment strategy in the AS/R system is not optimized, this will not be beneficial for the efficiency and therefore neither for the ROI.

1.5 Research questions

In this section, we discuss the research questions that are relevant for this research. Before we start with the sub-questions, it is relevant to mention the main research question we want to investigate during this research.

"How do we design an optimal storage assignment for the automated storage/retrieval system of Company X such that the travel time efficiency is maximized?"

In order to determine what products must be considered and why, and how many of these products are stored in the AS/RS, we need to collect data. Examples of this data include product data, production quantities, and forecast quantities. This must be in line with the expectations of stakeholders at Company X. Not only product data, but also characteristics and movement behaviors of the AS/RS must be collected from the supplier. The questions related to data collection are discussed in Chapter 2 and are:

1. What are the characteristics of the AS/R system placed in the warehouse of Company X?
2. According to the stakeholders at Company X, what are the key constraints, and objectives that must be considered for the AS/R system optimization?
3. How do we gather the most relevant SKU types and what are the key findings?

Hereafter, we need to gather sufficient information about different types of storage policies, space allocations, and characteristics of the correct AS/R system. In addition, we also want to gather information about mathematical models or algorithms that can be used to solve these kinds of problems. To be able to assess the performance of warehouses and AS/R systems, we also look at methods and techniques that are appropriate for that. In Chapter 3, the following questions are answered:

1. What types of theoretical frameworks, mathematical methods, or algorithms have been proposed for AS/R system optimization, and what are their strengths and limitations?
2. What different storage policies can be found in literature and what are the advantages and disadvantages of these policies?
3. What are the main points of interest for multi-deep AS/R systems and their implementation?
4. How is the performance of warehouses assessed and which methods or techniques can be used for that?

In order to determine how the AS/RS performs, we need to program a mathematical model with the characteristics and data collected in the previous section. For this, we answer the following questions in Chapter 2:

1. What assumptions do we make regarding characteristics of the model and the input values it uses?
2. How does the optimization model looks like and how do we implement the collected data?
3. What is the overall working of the simulation model?
4. What types of storage policies do we use in our model to determine the performance of the AS/RS?

Before we can formulate our conclusions and recommendations, we need to interpret and analyze the results from the mathematical model we have created in the previous section. Before we draw conclusions and recommendations, we check whether the solution can be improved further. This is all done in Chapter 5.

1. How do we improve the results found so far and which methods can be used for that?
2. What system utilization level yields the highest efficiency?
3. What are the results found and what do they say about the performance of the AS/RS?

Subsequently, in the last chapter of this research, we provide conclusions, recommendations, but also limitations of this research. We end with a section that includes suggestions for further research.

1.6 Scope

The optimization process of a warehouse is often complex and depends on several factors. These factors are again dependent on the type of industry the company is operating in. Examples of these factors could include size, location, flexibility, growth perspectives, and management wishes. With the upcoming influence of Industry 5.0 in the past years, automated processes have become more and more part of our daily lives. Industry 5.0 combines human intelligence and subjectivity with efficiency, precision of machines in industrial production, and artificial intelligence (Leng, et al., 2022). The automation of processes does not only increase efficiency of processes, but also the complexity.

At the moment of writing, Company X is building a new warehouse. Since the location and dimensions of the warehouse are already determined, strategic optimization falls outside the scope of this research. Besides, the warehouse is still under construction, which means that operational optimization is not very applicable either. Another thing that is already determined by Company X is that it includes an AS/R system in the warehouse. This AS/R system is covering approximately two-thirds of the maximum storage capacity of the entire warehouse, and determining where to store what pallet with raw materials is a complex problem. This can be classified as a tactical decision, and therefore we focus on tactical optimization in this research.

The goal of the research is to optimize the AS/R system in such a way that the efficiency of the system is maximized. This is done by positioning pallets in positions that are as optimal as possible. Things that we need to take into account are current stock quantities, product type, pallet heights, allergen characteristics, demand rates, and turnover ratio. With this product data we try to create an order list that is as representative as possible in order to create valid results. To create a valuable solution for Company X, we only focus on the storage assignment within the AS/R system, and not the entire warehouse.

1.7 Conclusion

This first chapter has introduced Company X and the sector in which it operates. Company X is building a new warehouse with a multi-deep AS/R system in it, and the goal of this research is to optimize this AS/R system. This is done by looking at production data, forecast quantities, and stock levels, but also at characteristics of the supplier of the AS/R system. To optimize this, we look at different storage policies and analyze which policy works best so that the total travel time is minimized. In the next chapter, we investigate further how the multi-deep AS/R system of Company X works and which data we are going to use to model our problem.

2 System and data description

In the upcoming chapter, we discuss and describe the way how data is collected and analyzed, in combination with the working and characteristics of the multi-deep AS/R system of Company X. These two points are discussed in Section 2.3 and Section 2.1 respectively. Last, the stakeholders vision and constraints are discussed in Section 2.2.

2.1 Characteristics of Company X's AS/R system

In this section, the characteristics of Company X's AS/R system are discussed. First, an explanation is given about how a general multi-deep AS/R system works. Thereafter, we mention some definitions in combinations with the corresponding characteristics. The following question is answered.

"What are the characteristics of the AS/R system placed in the warehouse of Company X?"

2.1.1 Working of a general multi-deep AS/R system

Before we actually describe the characteristics, we first explain the working of a general multi-deep AS/R system. Each system has its own configurations, but the general working of each multi-deep system remains the same. A multi-deep AS/R system has a high space utilization, since it uses less space for aisles. One of the disadvantages of this is that not all products are directly accessible. In such a multi-deep system, products are stored in a multi-deep lane behind each other, and storage or retrieval actions can be performed by one or multiple handlers. Xu et al. (2018) provide a graphical representation of a multi-deep AS/R system, which is presented by Figure 1 below.

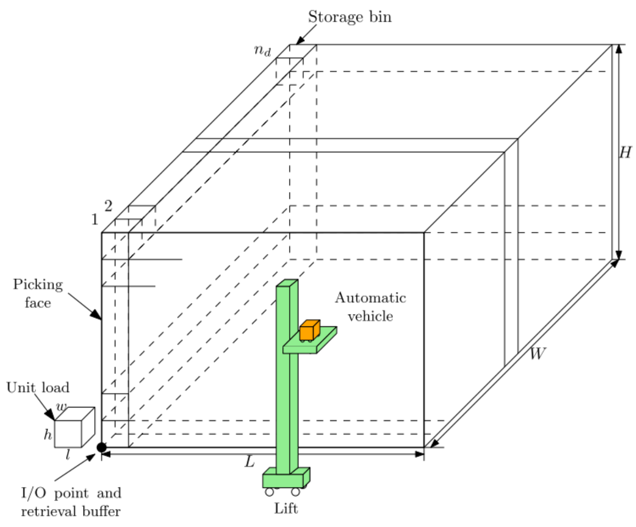

Figure 1: Multi-deep AS/R system (Xu et al., 2018).

As we can see in Figure 1, the vertical green pillar serves as a lift for the automatic vehicle, that is indicated by the orange color. This lift moves in front of the storage system, with length L and height H. Due to the ability of the lift to move horizontal and vertical, it can access all storage points that are directly faced at the first row. The number of rows is dependent on the width of the system, indicated by W. If a lane is filled with more than 1 product, not all products are directly accessible anymore. Only the first one remains directly accessible, and in order to access pallets that are placed deeper in the lane, the pallets in front must be removed first and stored in a special buffer area (Yu & de Koster, 2009). The advantage of having more storage space in multi-deep systems is at the expense of direct access of all products in traditional single-deep storage racks (Xu et al., 2018).

2.1.2 System Structure

To understand the multi-deep system of Company X completely, we elaborate on all the characteristics in the upcoming paragraphs. The definitions of certain AS/R system segments are discussed to clarify what is meant in the upcoming paragraphs, but also in later sections. The corresponding dimensions and velocities of shuttles are mentioned in this section as well.

Definitions and dimensions

Since we are considering a 3D multi-aisle AS/R system, this means that we have multiple floor levels. The first level differs from the other 4 levels, which is presented in Appendix B – Multi-deep AS/RS layout of Company X. Here, **Error! Reference source not found.** shows the top view of the first level, **Error! Reference source not found.** shows the top view of levels 2 and 5, and **Error! Reference source no t found.** shows the top view of levels 3 and 4. The top view of levels 2, 3, 4, and 5 look very similar, but there are some minor differences. The biggest difference lies in the heights between the levels. These different heights per level are shown in **Error! Reference source not found.**. The maximum storage height for levels 1 and 2 equal 1.5 meters, for level 3 1.8 meters, and for levels 4 and 5 2.25 meters.

The shuttles need to drive through aisles to store or retrieve products. 3 main aisles and 4 connection lanes are used for this. The three main aisles reach over the entire length of the AS/R system, to be able to access each pallet lane. These three main aisles have a length of 89.1 meters, and are shown in **Error! Reference source not found.**. The connection lanes provide the ability to move between main a isles, which are presented in **Error! Reference source not found.**. Each connection lane has a length of 10.7 meters, while the total length of the system equals 50,.85 meters. The main aisles and connection lanes are located at the same points for all 5 levels.

Considering **Error! Reference source not found.**, three main aisles separate 4 storage blocks. Of these 4 storage blocks, the middle two are accessible by both sides, and the other two are accessible by only a single side. The most southern block has a storage capacity of 8 pallets per storage lane, and the most northern block has a storage capacity of 11 pallets per storage lane. The most southern block has a storage capacity of 8 pallets per lane, and is only accessible from main aisle 1. The most northern block has a storage capacity of 11 pallets per lane, and is only accessible from main aisle 3. Preferably, products that need to be stored in large quantities, should be allocated to storage lanes that are accessible by only 1 side instead of 2 sides. This is due to the reason that an entire lane is occupied by the same product, with as result that this product is directly accessible. Smaller storage quantities are preferably assigned to the middle two blocks that are accessible from two sides. This minimizes relocating products that are blocking the way of the needed product.

Several pillars are used to keep the massive warehouse standing. A limitation of these pillars is that some capacity is lost in the AS/R system. In **Error! Reference source not found.**, the pillars are marked b y the red circle. Obviously, no products can be stored at these locations and all pillars are located at the same place for each level. These pillars obstruct some storage lanes, which means that some storage lanes cannot be used completely. The storage lanes are marked by the yellow striped pattern in **Error! Reference source not found.**. There are 4 different storage lanes, where storage lane 1 is present at the south side, and lane 4 is present at the north side. These storage lanes are separated by the 3 main aisles, and are visible in **Error! Reference source not found.** as well.

At some location the products must enter and/or leave the system. This is called the depot or I/O station. The AS/R system of Company X has two separate I/O points, both located at the top if seen from above. This is indicated by the two red circles in **Error! Reference source not found.**. The inbound s tation can also serve as an outbound station if the regular outbound is defective. The outbound station is not able to serve as an inbound station if the regular inbound station is defective.

The capacity of each level differs due to space restrictions for the inbound and outbound station. The total capacity of the entire AS/R system equals 8,680 pallets. This is divided over the 5 levels, which do not have the same capacity. Level 2 and 5 have the same capacity of 1,740 pallets, while level 3 and 4 both have a capacity of 1,736 pallets. Level 1 has the least capacity of 1,728 pallets. For level 1, the inbound and outbound have some automation machinery placed, which is at the expense of storage capacity.

If a product needs to be stored or retrieved, a shuttle is requested to perform this task. It is possible that an incoming order arrives with a height of more than 1.5 meters. This means that this product cannot be stored at storage levels 1 or 2. Storage levels 3, 4, and 5 are higher than 1.5 meters, and therefore this product can be stored at these levels. To reach these levels, an elevator is used as depicted in green in Figure 1. The difference is that the elevator in Company X's AS/R system is able to move upwards only, and not sideways. The location of the elevators is indicated by the yellow circles in **Error! Reference source not found.**. When a shuttle needs to perform a task that is not on the lowest l evel, it can start or end its task at the elevator position. This means that the shuttle does not have to travel all the way down to the inbound or outbound station, but can stay at its level. As a conclusion and overview, below in Table 1, we provide all the information described from the beginning of Section 2.1.2.

Part	Value
Storage positions in length	54
Storage positions in width	38
Storage levels	5
Total storage positions	8,680
Storage height levels 1 and 2	1.5 meters
Storage height level 3	1.8 meters
Storage height levels 4 and 5	2.25 meters
Storage capacity level 1	1,728 pallets
Storage capacity levels 2 and 5	1,740 pallets
Storage capacity levels 3 and 4	1,736 pallets
Number of shuttles	4
Number of main aisles	3
Number of connection lanes	4

Table 1: Overview of the AS/RS characteristics

Velocity of shuttles

The velocity of the shuttles has a significant impact on the performance of the system. The values described in this paragraph are provided by the supplier of the AS/R system. The speed of the S/R shuttles for each scenario in the AS/R system is listed in **Error! Reference source not found.** in the a ppendix. These values are equal for all 5 levels. The units of measure are meters per second (m/s) and meters per second squared (m/s^2) for speed and acceleration respectively.

In **Error! Reference source not found.**, the speed stands for the maximum velocity that the shuttle is a ble to reach. The acceleration happens with the stated speed. The deceleration of the shuttle happens with the same speed as the acceleration. The total time needed to reach the maximum speed can be easily calculated by dividing the difference in starting speed and maximum speed by the acceleration speed. For example, to reach the maximum speed of 1.4 m/s from a standstill with an acceleration speed of 0.5 m/s^2, we need 2.8 seconds. This principle is shown in Figure 2 below, where V_{max} indicates the maximum reachable speed, T indicates the total time, and t_p indicates the time of a transition point.

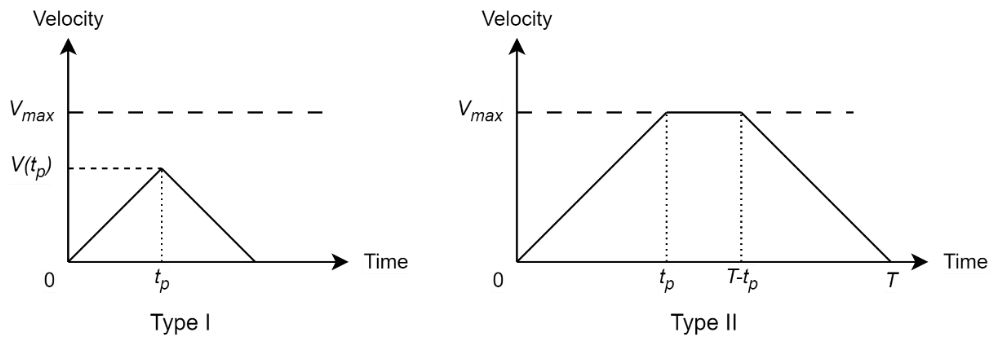

Figure 2: Speed model of the S/R shuttle (Yang et al., 2017)

The difference between the two types lies in the distance that the shuttle needs to travel. For type I, the shuttle travels such a short distance that it must decelerate before the maximum speed can be reached. At the point of deceleration, the time equals t_p and the velocity equals *V(*t_p*).* For type II, the travel distance is bigger which means that the shuttle reaches its maximum speed. The time that the shuttle moves at maximum speed equals *T-2*t_p. Besides, when a shuttle wants to move from connection lane 1 to connection lane 2, it has to cross main aisle 2. For safety reasons, the shuttle must slow down to a certain speed, which is also shown in **Error! Reference source not found.**. For type I, the total time TI (in seconds) that is needed to travel a certain distance X with an acceleration speed a can be calculated by using Equation (1). When the formula for type I is used exactly, is explained in Appendix E – Supplier-related information as well.

$$T_I = 2 * \sqrt{\frac{X}{a}} \tag{1}$$

For type II, where the maximum speed is reached, the total time T_{II} (in seconds) that is needed can be calculated by using Equation (2). Here, t_a indicates the time to accelerate to maximum speed, t_d indicates the time to decelerate from maximum speed to a standstill, and t_{max} is the time that the shuttle travels at maximum speed. Since we have the same constant acceleration as deceleration, we know that t_a is equal to t_d.

$$T_{II} = t_a + t_d + t_{max} \tag{2}$$

In our case, t_a and t_d can be calculated by using Equation (3) below. In this equation, V_{max} is the maximum reachable speed, and a is the acceleration speed.

$$t_a = t_d = \frac{V_{max}}{a} \tag{3}$$

In order to calculate t_{max}, we need the distance (in meters) that the shuttle travels at maximum speed. This distance is indicated by d_{max}, and is calculated by subtracting the acceleration and deceleration distance from the total distance. The distance that the shuttle travels during acceleration (d_{acc}) and deceleration (d_{dec}) is calculated by using Equation (4).

$$d_{acc} = d_{dec} = ut + \frac{1}{2}at^2 \tag{4}$$

Where,

u is the initial velocity in meters per second.

t is the acceleration or deceleration time in seconds. In our case, this equals t_a or t_d respectively.

a is the acceleration or deceleration in meters per second squared.

$$d_{max} = total\ distance - 2 * d_{acc} \tag{5}$$

$$t_{max} = \frac{d_{max}}{V_{max}} \tag{6}$$

An important assumption is made here, and that is that no main aisles are crossed. When the main aisles are crossed, the total distance is divided into more sub distances. These sub distances indicate the distance from a starting point to a main aisle, from main aisle to main aisle, or from main aisle to ending point. This is done to ensure that shuttles decelerate to the required crossing speed. If the shuttle is 1 or 2 pallet positions away from the main aisle it needs to cross, type I is used. Otherwise, type II is used.

The purpose of deceleration is to bring the shuttles to a standstill situation. This is necessary before the shuttles are able to perform certain tasks. Examples of these tasks include wheel set change, pallet storage or retrieval, and reshuffling. Changing the wheel sets happens each time when the shuttle wants to change its direction to left or right. The reason for this is that the aisles are not wide enough to make an actual turn with a square-shaped shuttle. The shuttle is able to change its wheel set automatically, without human intervention. The time needed for the abovementioned tasks is shown in **Error! Reference source not found.**.

Routing

The routing of the shuttles has a significant impact on the total travel time. The goal of the shuttles is to travel a route that is as short as possible, to reduce this travel time. Shuttles are able to move over main aisles and connection lanes at all times, but are also able to move through storage lanes. The only requirement for this is that the storage lane is empty, and the shuttle carries a pallet. If the storage lane is filled, and the shuttle does not carry a pallet, it is also possible for a shuttle to move through storage lanes. These storage lane movements are only possible over the width of the system. In a top view perspective, this means that the shuttles can only move from above to below, or vice versa. In this situation, the storage lane functions as a connection lane between the main aisles. Moving over the length of the system cannot be done by using storage lanes, but only over the main aisles. In totality, 4 shuttles will be moving through the system to perform the storage and retrieval requests. It is possible that these shuttles are performing tasks near each other, which could cause waiting times or delays. These assumptions are discussed in detail in Chapter 2, Section 4.1.

2.2 Stakeholders vision

In this section, several stakeholders were asked to give their independent vision on what is important to consider. The goal of this to understand and evaluate what is important to focus on. The following question is answered.

"According to the stakeholders at Company X, what are the key constraints, and objectives that must be considered for the AS/R system optimization?"

To answer this question, 4 stakeholders were asked independently, without each other's proximity. The job titles of these stakeholders differ from senior warehouse operator to warehouse and engineering managers. Their opinions are summarized in the paragraphs below.

2.2.1 Constraints

The first important constraint that must be taken into account is regarding allergens. Company X produces medical nutrition products for people in poor health. These people already have a weakened immune system, for who risks must be avoided at all times. The production of both powder and liquid products at Company X happens according to strict rules, to minimize unnecessary exposure to undesired ingredients or substances to end users. To accomplish this, Company X produces according to the Good Manufacturing Practice (GMP). GMP provides descriptions to which medicine manufacturers must comply with in their production process (Good manufacturing practice, 2024).

The 5 principles of GMP are also known as the 5Ps, and include People, Process, Procedures, Premises and Equipment, and Products. Since Company X produces medical nutrition products, GMP covers every area of production. This starts at the storage of raw materials, to ensure that these materials are well suited for production (Hole, Hole, & McFalone-Shaw, 2021). At Company X, the raw materials can be roughly classified into two categories. First, raw materials that carry allergens like milk, soy, gelatin, or cereals are classified as “allergens”. Second, raw materials that do not carry these types of allergens can be seen as “non-allergens”. The constraints are explained next by first giving the definition of the constraint, followed by an explanation.

1. The allergen storage constraint

The first constraint ensures that different types of allergens cannot be stored directly above each other. This minimizes the risks of cross-contamination and thereby potentially jeopardizing the end consumer. For example, it is not possible to store milk allergens above soy allergens, or cereal allergens above gelatin allergens. Obviously, the same types of allergens can be stored above each other, since these have the same allergen characteristics. Storing an allergen above a non-allergen is also not possible, but storing a non-allergen above an allergen is possible. Moreover, storing products next to each other does not have any limitations. This means that it does not matter whether an allergen is stored next to another allergen or a non-allergen.

2. Storage height constraint

In Section 2.1.2, we described that the system consists of 5 levels, that are not equal in height. A result of this is that it is not possible to store every product at each location. It is therefore important to take the height of the incoming pallet into account before a feasible location can be determined. In addition to that, IBC containers are able to carry liquid raw materials and must be stored at level 1 at all times. Last, each level of the system has a maximum capacity, which of course cannot be exceeded.

3. Accessibility constraint

The third constraint takes care of the accessibility of the ingredients. The AS/RS has multi-deep storage lanes, which means that not all products will be directly accessible in a fully occupied system. The stakeholders of Company X want that at least each type of product is directly accessible, and preferably that each batch number is directly accessible.

4. Cleaning constraint

An optional constraint could be to consider the cleaning process of the system. During the process of storing or retrieving pallets with powder ingredients, it is possible that some bags or boxes are unintentionally torn open. After some while it is necessary to clean certain parts of the system to comply with the high hygiene standards. This cleaning is done by employees and before this can be done the parts that needs cleaning must be empty. Therefore, it needs to be determined where these products need to be stored temporarily.

2.2.2 Objectives

The first objective that the stakeholders of Company X want to see is an optimal storage assignment strategy. Based on a representative list of SKUs that need to be stored, the total travel times of all shuttles must be minimized. How this representative list of SKUs is created is presented in the next section. When the total travel times are minimized, this decreases the overall utilization of shuttles. This eventually results in a higher system resiliency during peak moments.

The second objective that must be considered is the First Expired, First Out (FEFO) principle. The ingredients arrive at the warehouse and receive a batch number. This number includes size, ingredient name, allergen type, but also expiry date. Because of the latter, ingredients cannot be stored for an indefinite period of time. A result of this is that the ingredient with the shortest expiry date must be retrieved from the system. Hereby, the quality of the ingredients and the end product remains high and risks of unnecessary wastes is minimized.

Another objective that the stakeholders want to see is an analysis on the impact of increasing the utilization of the system. In the beginning, the system will not be entirely filled meaning that storage and retrieval targets are more easily achieved. As more pallets enter the system, less space remains for temporary storage of products. For example, if all 8,680 locations are occupied and a product at the end of the lane is needed, the products in front must be removed entirely from the system since no locations are available for temporary storage. At some level of utilization the travel times increase significantly, and the stakeholders want to know at what utilization level this happens.

2.3 Data description

Now it is known how the AS/R system of Company X works, and what is important to consider according to the stakeholders, we can gather appropriate data. This data is gathered from the Enterprise Resource Planning (ERP) System of Company X, with the assumption that the gathered data is correct. First, the data collection process is discussed, whereafter an analysis is given of the gathered data. In this section, the following question is answered.

"How do we gather the most relevant SKU types and what are the key findings?"

2.3.1 Selection of most relevant SKU types

First of all, in order to select the most relevant SKUs, we need to know what type of products are important to look at. Since only ingredients are planned to be stored in the system, and not finished products, we only look at raw materials. The senior warehouse operator suggested to use the current stock levels as a representative list. However, the managers explained that Company X does not produces all its products in the same quantities over the year. Using the current stock levels could give a distorted view of the reality, which means that we cannot simply use this.

As we cannot just simply use the current stock levels, a careful analysis needs to be done to determine the appropriate time interval to collect the data. The stakeholders argued that Company X is an innovative company, which resulted in a changing product portfolio over the past years. Production data of several years ago does not give an accurate representation of the current operations. An estimation was made that the production data of the last 365 days would give an accurate enough representation of what is produced nowadays.

Subsequently, it must be determined what type of data must be collected from the last 365 days. The purpose of the new AS/R system is to create more storage space for raw materials, and especially ingredients. Therefore, all the incoming pallets from any supplier that carry ingredients only were gathered, in combination with all the outgoing ingredient pallets. The incoming pallets were selected carefully in such a way that only the movements to the factory are taken into account. With this action we know for sure that the incoming pallets are used for production, and not for relocating or enlarging inventory. The result of this analysis was that 106 unique ingredients were found. This enables us to create two lists, of which the first one includes all the movements towards the factory, and the second one has listed the all the unique products exactly once, resulting in a list length that is equal to the number of unique products found. This gives us a clear view on which ingredients were needed more, and which ones less.

2.3.2 Analysis of collected data

In this section, an extensive analysis is made on the collected data described in the previous section. The number of movements for each ingredient were counted over the period from April 2023 until April 2024. These movement counts were sorted from largest to smallest, which gave us insight in the most needed ingredients and the least ones. In Figure F1 in Appendix F, the histogram depicts the frequency of movement occurrences in the list. The horizontal axis represents the intervals of movement counts, while the vertical axis displays the corresponding count of ingredients falling within each interval. The total number of movements were 37,743 pallets. A remarkable difference can be noted, as there are much more products with low movements than with high movements. Besides, a large difference between the highest ranked and second-highest product exist. The highest movement count equals 3,861, while the second highest count is 2,130.

Eventually, this total number is used to calculate the individual ingredient contribution to the entire movement count. The movement frequencies are ranked from large to small, from which we created Figure F2. The orange line in this figure shows the cumulative contribution to the movement count, where the number of ingredients is shown on the horizontal axis.

A further analysis can be done based on the allergen type and height of the products. Why this is important to take into account is discussed in Section 2.2. In Table 2 below, the types of allergens that occur in the original product list are presented according with their total number of movements. Besides, the number of products that carry the specific allergen in the unique product list is presented as well.

Type of allergen	Number of products	Movement count
Milk	407	180,273
Non	230	130,039
Soy	83	64,414
Cereals	12	1,744

Table 2: Number of products and movement count per allergen type

The majority of the movements of products can be categorized as milk-allergen ingredients, followed by the non-allergen ingredients. Soy-allergens are moved 64,414 times in the past 365 days, while ingredients with a cereal allergen were only transported 1,744 times. Finally, the height of the products was gathered as well. This height only included the product height, and did not take into account the height of the pallet. Since this is required for the AS/R system, we added the pallet height to each pallet in the list. The subsequent height occurrences are presented in Figure F3 in Appendix F**Error! Reference source not found.**. The horizontal axis presents the height intervals, and the vertical axis shows the occurrences.

2.3.3 Past stock levels analysis

Now we have determined the most important products based on the movement frequencies, a base stock level has to be determined for the AS/R system. This is needed to represent the current state of

Company X in the field of pallet storage as good as possible. A few assumptions are made here, of which one of them is that we pick the maximum stock level of each product over the last quarter. As introduced in Chapter 1, Company X is building a new production line at the expense of storage space, resulting in a higher throughput when this production line is finished. To represent this higher throughput, the maximum stock levels of the first 16 weeks of 2024 were used. This is visualized in Figure F4 in Appendix F. The total sum over all products equal 7,764. This means that this number of pallet positions will be occupied in the AS/R system from the beginning. The highest number of pallets we have to store for a single product equals 467, followed by 314 pallets for the second highest number. The lowest number of pallets that need to be stored is equal to 2.

With this information we are able to determine the amount of pallets we need to store that carry specific allergens. We use again the maximum stock level over the past 365 days, resulting in a total pallet storage for soy-allergen ingredients being 1,208 pallets, for milk 3,235 pallets, for cereals 30 pallets, and for non-allergen ingredients 3,291 pallets. These results are not very different from the results found in Table 2. Interestingly for the movement count, the ingredients carrying milk allergens are much more than for non-allergen ingredients. For the storage amount, the number of ingredients carrying milk and non-allergens are much closer together than for the movement numbers. On the other hand, the number of ingredients that are categorized as non-allergens are more than twice as much as milk-carrying products. In proportion, the milk ingredients are moved and stored in higher quantities.

2.3.4 Order list analysis

Another important element of the data collection part is an order list that reflects the current state of Company X as representative as possible, with as purpose to assess the performance of the AS/R system as good as possible. The movements from all suppliers to Company X during the past 365 days were analyzed again to determine the representativity. The number of pallet movements per week were summed, and displayed along with the average in Figure F5 in Appendix F. The average number of pallets moved per week is shown by the orange line and equals 724.

As discussed, Company X is expecting a growth in the near future, which means that we carefully need to select representative weeks. Another aspect that must be taken into account is that the production schedule happens according production cycles. According to a planning employee at Company X, approximately 4 cycles of equal length happen each year. Within each cycle, every product in the product portfolio of Company X is produced at least once. At the moment of the data collection, 16 weeks have passed in 2024. In Figure F5, the first value at the horizontal axis represents week 16 of 2024. The planning employee indicated that the first 13 weeks of 2024 is a representative production cycle. Since we are looking backwards in Figure F5, this production cycle lies between the horizontal axis values 3 and 16. For sake for clarity, this is indicated by the vertical red-dotted lines.

2.4 Conclusion

To conclude, Company X's AS/R system can be classified as a multi-deep system with 5 levels, reaching a total storage capacity of 8,680 pallets. Each level has its own maximum storage height, meaning that not all products can be stored in all places. Pallets that must be stored or retrieved are moved by shuttles. These shuttles drive autonomously and have limitations regarding speed and movements. Type I and Type II are two movement types that can be used by the shuttles. Preferably, shuttles move in 1 direction as long as possible, since this benefits the total travel time. Additional movements as changing the wheel set takes additional time, resulting in inefficient routing.

Besides, several stakeholders were asked about objectives and constraints that must be met. A major constraint takes into account the storage strategy of ingredients that carry allergens. These cannot be stored under or above each other, with as goal to minimize contamination risks. In addition, products of which the height exceeds the maximum storage height of a AS/RS level cannot be stored at that level. The same holds for the capacity, this cannot be exceeded for all levels. One of the objectives that must be achieved is an optimal storage assignment strategy. By using the FEFO principle, the products with the shortest expiry date are picked first, resulting in less wastes and higher product quality.

Last, data is gathered and analyzed with as goal to assess the performance of AS/R system. The pallets that were transferred to Company X for production purposes during the past 365 days were picked for this. This period ranges from April 2023 until April 2024. One of the employees from the planning department confirmed that this period would give a representative view of current production activities. Allergen types, heights, movement frequencies and lot sizes were determined to visualize the distributions of different data.

3 Literature Study

In this chapter, we discuss the first set of sub research questions. In Section 3.1 we discuss several methods that can be used to optimize AS/R systems, which is followed by the analysis of several storage policies in Section 3.2. Section 3.3 described the main points of interest of a multi-deep AS/R system, and eventually we discuss the performance of warehouses and AS/R systems can be analyzed.

3.1 Optimization methods for AS/R systems

Once we have now discussed several storage policies, methods, and AS/R types, we move on to the optimization methods that are used to optimize the AS/R systems. It is therefore that we answer the following question in this section:

"What types of theoretical frameworks, mathematical methods, or algorithms have been proposed for AS/RS optimization, and what are their strengths and limitations?"

Before we dive into the theoretical frameworks and methods, we first want to find out what we exactly want to know. As discussed in Chapter 1, Company X has already taken care of strategic decisions as location and dimensions of the warehouse. The supplier takes care of operational decisions as routing, but tactical decisions lies in the hands of the stakeholders at Company X. Tactical decisions include the allocation of products to functional areas, determination of manpower for operating the system, and the development of replenishment and order picking policies (Heragu, Du, Mantel, & Schuur, 2005).

The stakeholders at Company X want specifically a optimal storage assignment strategy for the AS/RS in the new warehouse. According to Yang et al. (2013), the storage location assignment problem (SLAP) is determining the way of assigning incoming products to storage locations, with as goal to maximize the order-picking efficiency. In the case of a multi-shuttle AS/RS, the locations for the retrieval requests must be determined as well. Ultimately, the efficiency of an AS/RS is maximized when SLAP is combined with the location assignment and storage/retrieval scheduling problem, in short LASRSP. Company X is producing according to a strict planning, and this means that the sequence of requests cannot be simply changed. Hence, the focus lies on SLAP, instead of on LASRSP.

3.1.1 Mixed Integer Linear Programming

Mixed Integer Linear Programming, also known as MILP, is a mathematical programming method used to optimize a linear objective function. This objective function is usually constrained by a set of linear inequalities, with some integer valued variables. MILP has a wide variety of application areas, but is one of the most usable methods for process scheduling problems (Floudas & Lin, 2005). Man et al. (2019) propose a bi-objective MILP method for optimizing a two-depot AS/R system. The first objective is to minimize the total travel time of the storage/retrieval system, and the second objective is to minimize the total tardiness of all the tasks.

The bi-objective problem described by Man et al. (2019) is solved by minimizing the two objective functions simultaneously, and the set of solutions correspond to the Pareto front. There are several methods to find these Pareto front solutions, among which the goal attain method, the weighted sum method, and the ϵ-constraint method. The latter method can be used to transform a multi-objective problem into a series of smaller single-objective optimization problems, with as goal to obtain Pareto optimal solutions. The downside of this ϵ-constraint method is that it is impractical and time-consuming to solve medium- and large-scale problems (Man et al., 2019).

One of the main advantages of the MILP method is the computational efficiency, since realistic problems often lead to extensive computational efforts. These efforts can be reduced by reformulating the constraints or by decomposition. The goal of reformulation is to generate models with a reduced number of binary variables, or tighter integrality gaps. Besides, the most widely employed strategy that is used to overcome the computational efforts is by using the so-called decomposition strategy. This strategy divides a large problem into smaller subproblems, which reduces the complexity and solution time of the problem (Floudas & Lin, 2005).

If we look at the limitations of MILP in general, but decomposition in particular, it often leads to suboptimal solutions. Besides, MILP formulations cannot take into account nonlinear effects, which could lead to inaccurate solutions. Second, risks exist due to the high-dimensionality of the problem, which is mainly due to the necessity to take into account all the time periods at once (Uranucci, 2018).

3.1.2 Simulation

Simulation is one of the most widely used operations-research and management-science techniques and is mainly used for the purpose of comparing alternative systems or configurations based on capacity and feasibility questions (Savory & Mackulak, 1994). Before these questions can be answered, a model needs to be created that gains some understanding about how the corresponding system behaves. This model is based on a set of assumptions, that usually take the form of logical or mathematical relationships. If these relationships are simple, mathematical methods such as algebra or calculus can be used to obtain exact information. However in reality, these relationships are too complex for an exact evaluation, which means that simulation must be used to answer the questions of interest (Law, 2015). According to Law (2015), some problems for which simulation is a useful and powerful tool include:

- Analyzing and designing manufacturing systems
- Analyzing supply chains
- Determining software and hardware requirements for computer systems
- Reengineering of business processes

As said, very complex situations cannot be described or evaluated by a mathematical model. As a result, simulation is often an appropriate method to investigate certain systems. A major advantage of using simulation is that the performance of an existing system can be estimated under a set of different operating conditions. These operating conditions can be controlled much better in an computer controlled environment, than compared to performing experiments with the real system itself. As a result of using these different conditions, the decision maker gets a more clear overview of which configurations of conditions meets the required performance. Last, a simulation model allows the decision makers to study the behavior of a system over a long time period, by using only a limited amount of time. On the other hand, it is also possible to study the behavior of the system under an extended period of time.

When deciding on whether simulation is an appropriate problem solving technique, the limitations of this method must be taken into consideration. A simulation study is based on a set of assumptions, and therefore only produces estimates of the system's true characteristics. Simulation models are therefore less appropriate for optimization, but more appropriate for comparing a fixed number of different system designs. Second, simulation programs are often expensive and it is a time-consuming process to develop a model with all appropriate assumptions. Last, a simulation study often generates a large volume of numbers. People tend to place a greater confidence in these outcomes than is justified. If the simulation model is not a good representation of the real system, the results provide inaccurate insights (Law, 2015). If we translate this to warehouses where demand rates are often stochastic, simulation is an appropriate method to estimate the consistent performance, especially when different zone sizes are considered (Silva, Roodbergen, Coelho, & Darvish, 2022).

3.1.3 Metaheuristics

Metaheuristics are first introduced by Glover in 1986, and is formally defined as an iterative process that guides a subordinate heuristic by intelligently combining various concepts to explore and exploit the search space. Learning strategies are employed to organize information, enabling the efficient discovery of (near) optimal solutions (Blum & Roli, 2001). Besides, metaheuristics are more powerful in the sense that they address larger issues, but also solve problems faster. Metaheuristics can be further classified as a group of approximate optimization approaches, which has seen a significant increase in popularity in the past two decades (Khan, Sinha, & Anand, 2023). In the field of AS/RS optimization, Bessenouci et al. (2012) use a taboo search and a simulated annealing approach for its flow rack AS/RS. Yang et al. (2015) use a Variable Neighborhood Search metaheuristic for LASRSP optimization in a multi-shuttle AS/RS. Another research by Fandi et al. (2022) incorporates a genetic algorithm for multi-shuttle AS/RS optimization, with a single S/R machine (Fandi, Kouloughli, & Ghomri, 2022).

Simulated Annealing

Simulated Annealing, in short SA, is a metaheuristic that can be classified as a memory-less algorithm. This means that the heuristic only uses information about the current state of search. At each state, a new solution is created, also called a neighbor, and checked whether the objective is better than the current objective found so far. In our case, the total travel time objective must be minimized. This process starts after creating an initial solution based on a starting heuristic. This can be a nearest-neighbor heuristic, a first-come-first-serve heuristic, or just a random assignment heuristic. When the neighbor solution is worse than the current objective, a probability measure decides whether the current solution is replaced by the neighbor solution either way. The probability measure can be defined by Equation (7) (Gogna & Tayal, 2013).

$$P = \exp\left(\frac{-\Delta E}{KT}\right) \tag{7}$$

Where ΔE indicates the change in energy (objective) between the best objective and the objective of the current neighborhood. K is the Boltzmann constant, and T is the control parameter at a given iteration, also known as the temperature. A common value for the Boltzmann constant K is 1, which we also shall use. This probability measure can allow a selection of a solution that is not improving the current solution, and thereby enabling the algorithm to escape from local optima (Gogna & Tayal, 2013). The pseudocode for the SA algorithm is discussed later, in Chapter 5. Moreover, the parameters of the algorithm must be chosen carefully. A higher number of iterations results in a higher running time, and a too low number of iterations might result in a solution that is not (near) optimal.

The SA algorithm is using both exploitation and exploration to find near-optimal solutions, where exploitation is focusing on the local search space to improve existing solutions. This is achieved by accepting solutions that have a better objective function. Exploration is used to explore a wide search space with as goal to avoid being trapped in local optima. This is done by occasionally accepting solutions with a worse objective function. The parameters of the SA algorithm influence the quality of the solution, and therefore need careful analyzation before SA can be used. For example, the Boltzmann constant determines the speed by which the algorithm is using exploitation to improve the solution. On the other hand, a higher number of iterations does not only cause a higher running time, it also determines the way the algorithm explores different solutions, rather than exploiting good solutions.

Variable Neighborhood Search

Variable Neighborhood Search (VNS), is a metaheuristic that has applications in various fields, such as communication, data mining, scheduling, and vehicle routine (Caporossi, Hansen, & Mladenovic, 2016). VNS can be compared to SA in such a way that both metaheuristic approaches can be used to avoid being trapped in local optima with a poor objective value. VNS is a metaheuristic that builds upon a local search method, that aim to improve the current solution. On the other hand, the perturbations allow the space of explored solutions to be extended. These two principles are often referred to as intensification and diversification (Caporossi, Hansen, & Mladenovic, 2016). Unlike traditional approaches that follow a specific trajectory, VNS explores progressively larger neighborhoods around the current solution. It moves to a new solution only if an improvement is found. This method keeps variables that are already optimal, leveraging them to find promising neighboring solutions (Hansen & Mladenović, 2001).

One of the advantages of using VNS is that it usually provides good solutions in a reasonable amount of calculation time. Besides, it has few parameters that must be tuned, unlike for SA, in order to get good results. The selection of the initial solution has a significant impact on the quality of the solution, but also on the computational efficiency. This is important, since often large-scale problems are involved. The key operation of VNS is the systematic change of neighborhood structure, in which it is possible that more than a single improvement solution is found in the neighborhood of the current solution. Choosing the best solution seems logic, but this requires a complete exploration of the neighborhood. The computational effort needed for this may be too large for the obtained benefit. This makes it more efficient to apply the first improvement that was found (Caporossi, Hansen, & Mladenovic, 2016).

Within VNS, it is possible to classify between different types. Examples of these types are General VNS, Reduced VNS, or Basic VNS. General VNS is more suitable for situations where the quality of the solution is preferred over computational effort. Here, the local search is replaced by a variable neighborhood search descent. On the other hand, it could be possible that the computational effort must be reduced, with as goal to handle large-scale problems better. In this case we speak of the Reduced VNS (Caporossi, Hansen, & Mladenovic, 2016). These forms of VNS differ somewhat between each other, but the basic concept of VNS remains the same. The following steps sketch a basic concept of the general working of a VNS (Yang et al., 2015):

1. Generating an initial solution.
2. Perform a local search with the current neighborhood.
3. If an improvement is found, update the current solution.
4. If no improvement is found, expand the neighborhood and perform a local search again.
5. Repeat until a certain stopping criterion is met.

Genetic Algorithm

The Genetic Algorithm (GA) is a popular metaheuristic research technique for solving optimization problems. GAs are a form of stochastic optimization that uses aspects of natural selection and genetics to solve the relevant optimization problem (Fandi, Kouloughli, & Ghomri, 2022). The GA starts with an initial set of solutions that are represented as chromosomes, which are often generated randomly or based on a heuristic. Each chromosome is evaluated on their performance, also called fitness. This fitness function indicates how close the obtained solution lies to the desired solution. The best chromosomes are picked and selected as parent chromosome for the next generation. The selection of parent chromosomes are based on some mechanism, which is more likely to choose better solutions since the probability of selecting these better solutions is proportional to the fitness. Once the parent chromosomes are selected, it replaces the previous generation. The GA iterates through the selection, reproduction, and replacement steps for a certain amount of generations or until a stopping criterion is met (Mirjalili, 2018).

By iterating over the process steps for multiple generations, the GA gradually generate better solutions to the optimization problem. The selection of parent chromosomes are based on a probability that is proportional to the fitness. Simultaneously, this process avoids selecting poor solutions, which leads to avoiding local optima. If good solutions are trapped in a local solution, it is possible that these are pulled out with other solutions (Mirjalili, 2018). The avoidance of local optima means that the entire solution space is searched, often resulting in finding a global optimum or a near-optimal solution. In general, GAs are suitable for solving large-scale optimization problems. The algorithm is applicable in a wide variety of problems, among which strategy planning, robot trajectory planning, and TSP and sequence scheduling (Sivanandam & Deepa, 2008).

The Genetic Algorithm has a limitation in the field of real time applications. For most optimization problems this is not a big issue, but if we look at real-time control systems or high-frequency trading, using GAs is not suitable. Another disadvantage of using GAs is that it usually takes long to converge to the optimal solution. These convergence times cannot be predicted either. Besides, the chosen parameters heavily influence the performance of the algorithm. Examples of these parameters are population size, mutation rate, and selection method. Choosing the right parameters involves experimentation and fine-tuning, which is sensitive to errors and thus the performance of the GA (Sivanandam & Deepa, 2008).

3.1.4 Conclusion

Several optimization methods are discussed, of which MILP is the only analytic one. Analytic research techniques can be used to solve optimization problems exactly. The downside is that it takes a large amount of computational effort to accomplish this, especially if the size of the problem becomes larger. Simulation is more efficient when dealing with large-scale problems. This method can also analyze existing systems under different operating conditions well. Since the design of the AS/R system of Company X is already determined, in combination with the fact that it is a large-scale problem, simulation is an appropriate method that can be used for the SLAP optimization. In addition to that, metaheuristics can handle these larger problem instances efficiently as well. Examples of metaheuristics that are used for previous AS/RS optimization are Simulated Annealing, Variable Neighborhood Search, and Genetic Algorithm.

3.2 Storage policies

In this first section we discuss different types of storage policies. There are multiple of these policies treated in literature, and each of these policies has its own application area. This means that a policy could be more convenient for pallet warehouses, while the other is more convenient for AS/R systems. We want to find out which policies there are and which advantages and disadvantages these policies carry. Therefore, the following question is answered in this section.

"What different storage policies can be found in literature and what are the advantages, disadvantages, and challenges of these policies?"

3.2.1 Random

The random storage policy is one of the most simple forms of storing products, since it does not use any information about the product. It ignores the product characteristics and time characteristics, and to each available location it assigns an equal probability of having an incoming load (Roodbergen & Vis, 2009). Storing products randomly has several advantages over the other policies, since it needs the least amount of space to store all the products. This smaller space reduces the travel cost to a particular location. On the other hand, using a random storage policy also results in a constantly changing location of the products. A result of this is that operators need more time to retrieve a product due to searching time. This searching time increases for fast-moving products, since these products are needed more frequently. If we translate this to AS/R systems, this means that the number of storage and retrieval transactions per unit time are not effectively utilized, and therefore not optimal. In order to use this policy in an efficient way, it is necessary to continuously keep track of inventory locations. This requires almost always a fully automated system such as an AS/R system with automated cranes (Goetschalckx, Storage Systems and Policies, 2012).

3.2.2 Dedicated

For the dedicated storage policy, each product type is assigned to a fixed location, and replenishments of this product always occur at this same location (Roodbergen & Vis, 2009). This policy is easily implementable by companies of all sizes, and does not need any expensive IT investments (Fumi, Scarabotti, & Schiraldi, Minimizing Warehouse Space with a Dedicated Storage Policy, 2013). Disadvantages of it are the high space requirements, due to the fact that the pre-defined locations are reserved for certain products, even if these products are out of stock. A perfect example of this occurs when a company faces a lot of seasonal sales. In that way, storage locations remain open for a period during the year, while during the selling season these locations are filled. Especially when the selling season is short, this is a very inefficient policy (Fumi, Scarabotti, & Schiraldi, 2013). When each location can only be occupied by a single product at a time, we can define M_{pt} as the number of storage locations that are used by item p at time t. Subsequently, M_{DED} is the overall number of locations required to allocate all the products in the warehouse:

$$M_{DED} = \sum_{p} max_t\{M_{pt}\} \tag{8}$$

This policy clearly does not minimize the number of storage locations needed. M_{DED} is usually seen as an upper bound for the number of storage locations needed (Fumi, Scarabotti, & Schiraldi, 2013). Another disadvantage can be found in the low space utilization of this policy. The reason for this is that for each product enough space must be reserved to accommodate the maximum inventory level that could occur (Roodbergen & Vis, 2009). If we compare the dedicated storage policy with the random storage policy, the random policy requires less storage space than the dedicated policy, provided that the volumes and the frequency of storage and retrieval operations are the same. In addition, this policy is advantageous in the field of data-handling efficiency, due to the fixed addressing of storage items (Lee & Elsayed, 2005). Most advantages for the dedicated policy, such as matching the layout of stores and locating heavy products at the bottom, are related to non-automated order-picking areas and are thus not as interesting for AS/R systems (Roodbergen & Vis, 2009).

3.2.3 Class-based

According to Roodbergen (2012), the class-based storage policy is the best of two worlds. This policy divides the products in different classes, and each class is then assigned to a dedicated location in the warehouse. Within that dedicated location, storage happens randomly. The classes are determined based on some performance indicator, for example, demand frequency or volume. These performance indicators are described in a later paragraph in this section. Products with high demand frequency are called fast-movers, while products with low demand frequency are called slow-movers. Fast-movers are categorized as A-items, and the next fastest moving category is called B-items, and so on (Roodbergen K. J., 2012). A-items usually contain only about 15% of the product portfolio, but it contributes to 85% of the total turnover (De Koster, Le-Duc, & Roodbergen, 2007). Typically, only 3 classes are used, and in that case we refer to ABC-storage.

One of the main advantages of the ABC-storage policy is that fast-moving items are stored near the In/Out (I/O) point. This increases the efficiency of order picking and storing, since more frequently needed items can be reached faster, resulting in a lower traveled distance in the long run. Moreover, simple implementation, manageable maintenance, and the ability to cope with product mix and demand variability are other advantages of the class-based storage policy (Bahrami, Piri, & Aghezzaf, 2019). For the ABC-storage policy, there are two commonly known methods for zone positioning, namely within-aisle and across-aisle. The difference is shown in Figure 3 below. Clearly, the A-items are stored closest to the depot, while the slowest moving items have the greatest distance to the depot.

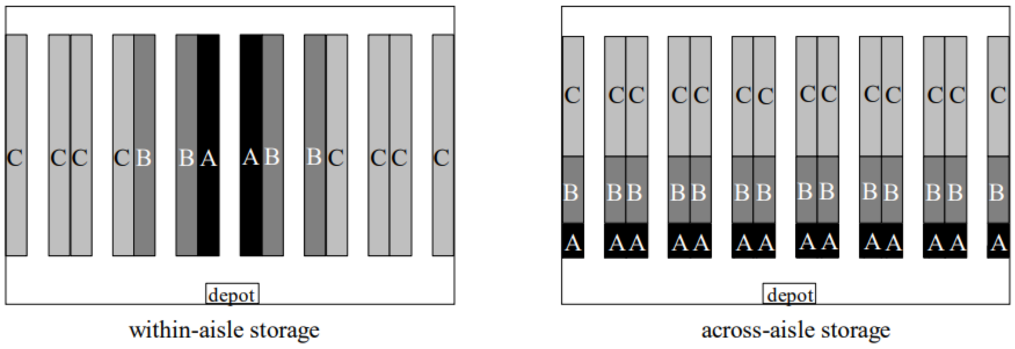

Figure 3: Illustration of within-aisle storage and across-aisle storage (De Koster, Le-Duc, & Roodbergen, 2007).

Roodbergen & Vis (2009) discussed in their paper that class-based storage can be applied to several situations, among which 3-dimensional storage problems and stochastic environments. Furthermore, if the class-based policy is compared to the dedicated policy, the former does not need a fully sorted list of SKUs and no comprehensive administration. Therefore, the class-based policy is easier implementable than for dedicated policy.

The number of classes is not always limited by 3, because more classes could give additional gains with respect to travel times. Van den Berg and Gademann (2000) performed a simulation study in which they simulated an automated storage/retrieval system. The conclusion of this study was that 6 classes, combined with the nearest-neighbor rule for selecting open locations, performed well compared to other policies. It should be noted that in order to be able to store an incoming load in the designated class region, it is necessary to have empty slots available. An increase in the number of classes therefore results in an increase of space requirements. Thus, class-based storage needs more rack space than the random storage policy, which is therefore an disadvantage of the class-based policy (Graves, Hausman, & Schwarz, 1977).

Implementing the class-based policy is not so obvious as implementing the random storage policy. Especially when considering AS/R systems, 3 major decisions are faced during the implementation of the class-based policy. The first challenge arises when the decision maker needs to determine the number of classes, also called zone divisioning. Second, the decision maker needs to determine the number of products that are assigned to each zone, also known as zone sizing. The last challenge that arises for decision makers is zone positioning, which means that the location of each zone must be determined (Roodbergen & Vis, 2009). In the next paragraphs, we discuss several strategies that contribute to solving the problems just described.

Cube-per-Order Index
The Cube-per-Order Index (COI) is a well-known storage strategy which was introduced by Heskett in 1963. The COI is a ratio, where items with a lower ratio are assigned to locations nearest to the I/O point. The ratio is based on the required storage space to the order frequency of that specific SKU (Caron & Marchet, 2010). Formally, Manzini (2012) provide a formula for the COI, and is given by Equation ((9).

$$COI_{i,T} = \frac{v_{i,T}}{\sum_{\text{order } j \in T} x_{ij}} \tag{9}$$

Where $v_{i,T}$ equals the average storage level of SKU i in time period T, and

$$x_{ij} = \begin{cases} 1 & \text{if item } i \text{ occurs in order } j \\ 0 & \text{otherwise} \end{cases}$$

It is important to mention that the COI strategy does not perform equally for single-command cycles as double command cycles. The COI is an item oriented strategy that is excellent for minimizing the order picking travel time when using single command cycles only. This means that the order picker starts at the I/O point, travels to the desired location and retrieves the desired product, and then travels back to the I/O point. If dual command cycles are used, i.e. storage and retrieval happens between two consecutive I/O visits, the COI approach performs worse than order oriented strategies (Schuur, 2015).

Turnover time
The second strategy that can be used is by looking at the average turnover time of the products. The turnover time is also known as the dwell time, which refers to the duration that a product stays in the shelf. In other words, this is the time between two successive replenishments. With this strategy, all the products are ranked based on frequency of requests and all locations are ranked from best to worst. The locations that are close to the I/O point are higher ranked than locations that need more time to reach (Manzini, 2011). Because the product portfolio of a warehouse is not fixed, it might happen that demand shifts occur. It is in that case necessary to relocate the products on time, to guarantee the efficiency

The average dwell time can be estimated by the warehouse manager, but it is also possible to calculate it. One necessary aspect for a successful implementation of this strategy is that the turnover frequencies need to be known beforehand (Manzini, 2011). Since we know that the time between two successive replenishments, i.e. the cycle time, can be determined by dividing the order quantity of SKU i (Q_i) by the demand of SKU i (D_i). In practice, the order quantity is often determined by the Economic Order Quantity (EOQ) formula. Subsequently, we can determine the average dwell time for SKU i (T_i) by using Equation (10).

$$T_i = \frac{Q_i}{2 * D_i} \tag{10}$$

Duration-of-Stay
The Duration-of-Stay strategy is also abbreviated as DOS and classifies every product according to their estimated duration of stay. The aim of DOS is to have minimum storage requirements while at the same time minimize the labor costs as well. The products with the lowest expected DOS are placed closer to the I/O point than the products with a higher expected DOS (Manzini, 2011). If we compare this method with the random strategy for example, the DOS method needs significantly more data for a successful implementation (Goetschalckx & Ratliff, 1990).

The performance of the DOS method is directly related to the balance of the warehouse. The balance can be described as the ratio between incoming and outgoing SKUs. When a warehouse is perfectly balanced, this ratio equals 1. The DOS method is more efficient in warehouses that are more balanced. On the other hand, if a company deals with high demand fluctuations, the system could have problems with the adaptation on time (Curry, 2013).

Family-grouping
The family-grouping strategy is also known as the correlated strategy, assigns items to a particular family if they are ordered together more frequently. As a result, it is convenient to place the products that are often ordered together in the same area. The location of each family is to be determined, dependent on a combination of the characteristics of all the products in the group. The space requirements for grouped storage policy like this one requires more storage space than the random storage policy. One of the disadvantages of family-grouping lies in the data collection procedure that is necessary to determine the statistical correlation between products. For example, the frequency at which products appear in the same order can be used, and should be known or at least predictable (De Koster, Le-Duc, & Roodbergen, 2007).

The two types of family-grouping strategies that are most frequently used are the complementary-based method, and the contact-based method. The complementary-based method consists of roughly of 2 phases. The items are clustered based on correlated demand in the first phase, and the items in the same cluster are located as close together as possible in the second phase. The contact-based method is very similar to the complementary-based method, only now the contact frequencies are used for clustering items. Contact frequencies are obtained by determining the number of times that item i is picked after item j, or item j after item i (De Koster, Le-Duc, & Roodbergen, 2007).

3.2.4 Greedy Algorithm
In Section 3.2.1, we discussed the random storage policy, and this policy only works in a computer-controlled environment. When we do not have such an environment and the operators are responsible for storing the products, we probably end up with the Greedy Algorithm, also known as the closest open location storage policy (De Koster, Le-Duc, & Roodbergen, 2007). In theory, under the closest open location policy the closest, available location to the I/O point is picked for storage (Goetschalckx, Storage Systems and Policies, 2012). Park (1987) concludes that in an AS/R system, the locations near the I/O point have the highest utilization rates. This leads to racks that are full around the depot and gradually become more empty towards the back. Besides, the closest open location policy performs similar compared to the random storage policy if the products are moved by full pallets only (De Koster, Le-Duc, & Roodbergen, 2007).

Advantage is easy to implement, no extensive data analysis and preparation are needed. In addition, the order pickers need to travel less distance to store the next product. This leads to an increase in efficiency, while at the same time increasing the order fulfillment speed and productivity. A disadvantage of this policy is comparable to the random storage policy, since comprehensive administration systems are needed to keep track of product locations. Besides, an effect of this policy is that warehouse activities are concentrated around the I/O point. This can lead to congestion and overcrowding. Overall, the closest open location policy benefits in terms of ease of implementation, but has limited flexibility (De Koster, Le-Duc, & Roodbergen, 2007).

3.2.5 Conclusion
According to Xu et al. (2018), the random storage policy is most frequently used in AS/R systems. An advantage of this is that has a high operating efficiency, and human order pickers do not spend a lot of time on searching a product when using such an automated system. The dedicated storage policy has a high data-handling efficiency, but also requires more space than randomized storage. Class-based storage is a combination between randomized storage and dedicated storage. Products are assigned to

classes based on some indicator, which could be the Cube-per-Order Index, Duration-of-Stay, Turnover time, or Family-grouping. Within that class, randomized storage applies. Eventually, when using the Greedy Algorithm the efficiency can increase, but it can also lead to congestion and overcrowding at popular locations.

3.3 Points of interest of a multi-deep AS/R system

As we have discussed in Chapter 1, Company X is building an AS/R system at the moment of writing. In Chapter 2, we have discussed the AS/R system of Company X in detail. The type of AS/R system can be classified as a multi-deep system with multiple levels. In this section, we find out what is already known on these types of AS/R systems. The following question is answered in this section.

"What are the main points of interest for multi-deep AS/R systems and its implementation?"

According to Xu et al. (2018), several decisions must be taken for a successful implementation of a multi-deep AS/R system. Roughly, these decisions can be divided into two categories, namely system structure and operating policies. The system structure refers to the size of the system, or the number of robots or shuttles. On the other hand, operating policies are not predetermined and must be evaluated. Examples of these operating policies include the type of command cycle, dwell point strategy, and the type of storage policy (Xu et al., 2018).

3.3.1 System structure

The design of an AS/R system is a crucial step, since the goal of the AS/R is to handle current and future demand requirements efficiently, while at the same time avoiding bottlenecks and overcapacity. The AS/R systems have inflexibility of layout and it is therefore important to design the system right at once (Roodbergen & Vis, 2009). The determination of size is an example of a system structure decision, and this is not unimportant since the size can significantly affect the operational efficiency of a warehouse (Yang et al., 2017). Determining the optimal rack dimensions involves several objectives, for example, the minimization of expected travel times, the minimization of investment costs, but also to anticipate on expected future growth.

When considering an AS/R system, and particularly a shuttle-based AS/R system, warehouse managers need to determine the number of S/R machines. For a shuttle-based AS/R system, this involves the determination of the number of shuttles. These decisions must be taken carefully, since these shuttles have high investment costs and account for a majority of the total investment costs. If a warehouse manager decides to use more shuttles, this increases the capacity on the one hand, but it increases the probability of collisions and delays on the other hand. The priority of shuttles is to avoid collisions, but to accomplish this, other shuttles need to wait until the transaction of the incoming shuttle is done. This situation is also called blocking delay and causes significant inefficiency in the system (Ha & Chae, 2019).

3.3.2 Operating policies

Operating policies, also called control policies by Vasili et al., (2014), are methods that determine how a AS/R system performs its tasks. A major operating policy decision is the type of storage policy that should be used in the system. According to Xu et al., (2018), a class-based storage always outperform the random storage policy in a multi-deep AS/R system, based on the expected travel time. Since the operating policies are already discussed extensively in Section 3.2, we refer to that section for further details.

Yu & de Koster (2009) discuss optimal zone boundaries for a multi-deep three-dimensional (3D) storage system, which is also called a compared or super high-density storage system. The advantages of such systems is that the efficiency of order picking is increased, while using minimum floor space. The optimization process for a 3D system is more complex than 2D AS/R systems, due to the larger number of variables involved. The authors show that zone dimensions significantly influence the travel time of the S/R machine. The results are compared with a random policy and show that travel times of the machines are significantly reduced (Yu & de Koster, 2009).

Another point of interest for an efficient use of a multi-deep AS/R system, but also a AS/R system in general, is the dwell point position. The dwell point can be described as the location where the S/R machine is waiting after it performed a storage or retrieval request. The goal of this relocation is to anticipate on the next request, which can be a storage or retrieval request, based on the relative rates of occurrence of each request. According to Meller & Mungwattana (2005), the dwell-point location significantly influences the performance of AS/R systems. However, under a high system utilization, simple dwell-point location policies do not influence the overall performance of the system in a negative way, compared to more complicated policies (Meller & Mungwattana, 2005). According to Roodbergen & Vis (2009), the optimal dwell-point position is the input station, given that the probability of having a storage request after an idle period is at least 0.5.

A multi-deep shuttle-based AS/R system like the one of Company X is able to operate according to a single command cycle or a dual command cycle. In a single command cycle, the crane performs only one storage or retrieval request between two consecutive I/O visits, while a dual command cycle performs both storage and retrieval between the two I/O visits. The storage cycle time is defined as the time that is needed to travel from the I/O point to the storage location, perform the storage action, and travel back to the I/O point again. The retrieval cycle time can be described in a similar way, but now the retrieval action is performed. Clearly, the total time that is needed to perform all storage and retrieval requests decreases when only dual-command cycles are used. Sequencing rules can be used to minimize the total travel time (Roodbergen & Vis, 2009).

3.3.3 Conclusion

When looking at designing AS/R systems, but multi-deep AS/R systems in general, roughly two categories can be distinguished. System structure involves the physical design of the system, including size, number of aisles, and dimensions. Although most of these decisions are definite, the number of shuttles can be changed, which is also a system structure decision. On the other hand, operating policies determine how the system performs its tasks. Zone boundaries, storage policies, dwell-point position, and command cycle strategies can be classified as operating policies.

3.4 Assessment of performance

In the previous sections several strategies were discusses that can be used to improve the efficiency of warehouses. When decision makers decide to implement one of the mentioned strategies, how do they actually know whether it actually improves the performance of the warehouse? Researchers have used several methods, ranging from short-term or long-term objectives, to the way how to measure these objectives and the type of warehouse system (Staudt et al., 2015). In this section the following question is answered that helps the decision makers by determining the warehouse performance.

"How is the performance of warehouses assessed and which methods and techniques can be used for that?"

Before the methods are discussed, we first focus on what is actually meant by performance. Performance is related to the way how work is done, which can be good or bad. Performance measurement is used to quantify whether work is good or bad, and does this by looking at effectiveness and efficiency of a task or activity. If things are not working as they should, performance measurement is one of the techniques that help to identify the causes of this incorrect performed work. Reasons for performing performance measurement are to improve performance, avoid inconveniencies, and maintaining quality (Liviu, Ana-Maria, & Emil, 2009).

Warehouses have at least one major functionality and that is to store products. Additional activities of warehouses include organizing transportation to customers, timely shipments of orders, or value-added activities. The performance of warehouses has therefore multiple dimensions. Usually, this performance is measured by ratios of input and output factors (De Koster, 2008). Input factors are the resources that are needed to achieve the output. Examples of these factors are investments in buildings and IT infrastructure, but also the process management. Output factors are the amount of shipped orders, quality (i.e. error-free and on-time delivery), but also flexibility. The flexibility is the ease by which a warehouse is able to cope with changes provided by the customer (De Koster, 2008).

According to Park (2012), a lot of warehouses use key performance indicators (KPI) to manage the functioning and productivity. The most commonly used KPIs for warehouses are productivity, operation costs, order accuracy, or order time. Based on these KPIs, a manager is able to carry out a warehouse performance gap analysis. This is a graphical representation of the performance of the warehouse in each dimension, compared with the industry best. The goal of this analysis is to point out strong and weak points in the warehouse (Park, 2011).

Staudt et al., (2015) provide 4 indicators to assess the performance of warehouses. These indicators are time, quality, costs, and productivity, and can be considered in a broad sense. Literature provides different measures that contribute to the performance of the indicators. For example, the time KPI is influenced by the order picking time, queueing time, dock to stock time, and equipment downtime. For a AS/R system, the turnaround time is a convenient indicator for the performance. The turnaround time can be defined as the time between the receival of a request and the actual completion of the request (Lee H. F., 1997).

The second performance indicator that a warehouse manager could use is quality. Some indicators include order fill rate, storage accuracy, stock-out rate, and cargo damage rate. However, in literature the emphasis lies at delivery on time and customer satisfaction. Since the AS/R system at Company X is part of a bigger warehouse, the quality of this system contributes to the quality of the entire warehouse. Indicators like order picking accuracy are less relevant for automated systems, since they do not involve human handling during the storage and retrieval process.

Costs is the third performance indicator, of which the least can be found in literature. This can be linked to that the evaluation of operational-level performance is usually based on non-financial indicators. Examples of cost indicators are inventory costs, order processing costs, labor costs, and maintenance costs (Staudt et al., 2015). Investment costs are left out of consideration, since Company X is already building the system at the moment of writing.

The last performance indicator is flexibility, and means how well some resources are combined and used to achieve desirable results, or the level of asset utilization (Staudt et al., 2015). Some productivity indicators are comparable between different situations. For example, even if the number of working hours per year in a country differs from another country, it is still possible to compare the labor productivity between these countries. Other examples of the flexibility indicator are throughput, turnover, transport utilization, and shipping productivity (Staudt et al., 2015).

The above mentioned performance indicators can be classified as “hard” metrics, which treats quantitative measures. On the other hand, “soft” metrics can be used as well and include qualitative measures like the perception of warehouse managers on customer satisfaction and loyalty (Kusrini, Novendri, & Helia, 2018). The downside of these soft metrics is that it is much harder to measure and thus determine how good a warehouse actually performs.

To conclude, if we relate the abovementioned indicators to our research problem, where we want to find the optimal location for pallets in a AS/R system, it would be convenient to use at least the travel time indicator. One of the reasons for this is that the travel time is influenced by the routing distances of the shuttles, in combination with the waiting times of shuttles. As the total travel time of all shuttles decreases, the efficiency and throughput of the system will automatically increase.

3.5 Conclusion

In this chapter, the focus lied on what past literature contributed to optimization methods, storage policies, and performance assessments in the field of AS/RS, but also warehouses in general. Mixed Integer Linear Programming is an analytic method that is used to solve the problem exactly, but has as limitation that it is not an efficient method for solving large problems. Simulation is more efficient when dealing with large-scale problems. In addition to that, simulation is an appropriate method when performances under different operating conditions must be analyzed. Because of this reason, simulation is a suitable method for our research. Metaheuristics are able to handle large-scale problems as well, and have as additional benefit that some of them are able to escape local optima. Generally, Genetic Algorithms require a larger population size and more iterations to converge to a near-optimal or even optimal solution. Hence, Simulated Annealing and Variable Neighborhood Search are metaheuristics more suitable for our research.

Storage policies are examples of operating policies and each of them comes with advantages and limitations. The random storage policy uses less space to store all products, but requires higher travel times of the S/R machine. The Greedy Algorithm is simple to use, but requires computer systems to keep track of product locations. Dedicated Storage uses a predetermined location for each products, which has as result that order pickers do not need much time to search for the product. Class-based storage assigns products to classes, assigns a location to each class based on the product characteristics within that class, and stores products randomly in that class. Different strategies can be used to assign products to classes, namely Cube-per-Order Index, Turnover time, family-grouping, or Duration-of-Stay.

When designing an AS/R system in general, but in particular a multi-deep AS/R system, system structure decisions and operating policies decisions arise. System structure decisions involve the size of the system, dimensions, number of aisles, and number of robots or shuttles. Operating policies include storage policies, dwell-point position, and type of command cycle. Last, warehouse managers need to be able to assess the performance of warehouses. So-called KPIs can be used for that, which for warehouses include time, quality, costs, and productivity. Specifically for this research, we look at the total travel times of shuttles. To put this into perspective, Table 3 provides an overview of research on AS/RS control methods and system configurations, with our research included.

	Controls						Configuration						
Authors	Storage assignment	Storage restrictions	Scheduling	Simulation	(MI)LP /DP	Meta-heuristics	System configuration	Unit-load	Single-deep	Double-deep or Multi-deep	Cranes	Multi-shuttle	Multiple I/O
(Randhawa & Shroff, 1995)	×		×	×				×	×		×		
(Yang et al., 2015)	×		×		×	×		×	×			×	
(Bessenouci et al. (2012)			×	×		×		×	×		×		
(Yang et al., 2013)	×		×		×	×		×				×	
(Yang et al., 2017)					×		×	×		×	×		
(Kouloughli & Sari, 2015)				×	×		×		×		×		
(Zhou & Mao, 2010)	×			×		×		×			×		
Our Research	×	×		×	×	×		×		×		×	×

Table 3: Overview of research on AS/RS control methods and system configurations

4 Model description

In this chapter, the focus lies on describing the optimization model in combination with the used data and strategies. Specifically, this is discussed in Section 4.2, while the assumptions regarding the characteristics of the model and the input data it uses are discussed first, in Section 4.1. In Section 0, we discuss several storage policies that can be used in the AS/R system.

4.1 Assumptions

In the upcoming paragraphs, the model assumptions are discussed. The goal is to create a model that is as representative as possible, but considering the fact that the entire AS/R system has not been built yet, it is necessary to make some assumptions. The following question is answered in the paragraphs below.

"What assumptions do we make regarding characteristics of the model and the input values it uses?"

Stochasticity of errors

In the current warehouse area located at the production plant, Company X is already working with automated guided vehicles (AGVs). In the new AS/R system, 4 shuttles will move and operate independently without the intervention of warehouse operators. This new situation is to some extent comparable with the old situation, in the sense of not using humans to transport pallets between locations. The experience of Company X is that these current AGVs malfunction every now and then, and although a different type of vehicle is used in combination with a different supplier, it remains possible to experience malfunctions or failures of these shuttles in the new AS/R system. We assume that the malfunctions of these shuttles occur in such a low rate, that it does not influence the performance of the AS/R system in the short term. In the long term, this would be very different. However, this is not the purpose of this research.

Another error that could occur is the malfunction of the outbound station. As discussed, the AS/R system does not have a single I/O point, but an inbound station that is separate from the outbound station instead. According to the supplier, it is possible to use the inbound station as an outbound station when the regular outbound station is not working. Again, we assume that the failure rate of the regular outbound station is too low to influence the overall AS/RS performance on the short term.

Employee availability

The performance of the AS/R system is not only dependent on the system itself, but also on the way it is used and the operations around it. For example, if several pallets are requested from the system, the shuttles transport the pallets to the outbound station. At this station, a warehouse operator must pick up the pallet with a forklift truck and move it to truck loading area. If there is no warehouse operator available to pick up the pallet at the outbound, and the shuttles are continuing with transporting pallets to this outbound station, the pallets are stored at the elevator lane. This elevator lane has a capacity of 5 pallets, which means that if this elevator lane is completely full, the shuttle needs to wait. We assume that there are enough employees available such that the total number of pallets waiting for pickup does not exceed 5. This is a realistic assumption, since the outbound station of the AS/R system is located near the truck loading area.

Another point that is not unimportant to mention here is that the employee availability also indirectly affects the pallet waiting times at the elevator. The outbound elevator lane has a capacity of 5 pallets, which means that if the elevator lane is fully occupied, pallets need to wait before they can enter the elevator.

Shuttles
After having several conversations with the supplier of the new AS/R system, the supplier came to the conclusion that the requirements with respect to throughput rate and number of requests per hour is low enough to keep the shuttles at their base levels. This means that shuttles do not have to move between levels to meet the required throughput rate. However, the total number of shuttles equal 4, while there are 5 storage levels. As a result, one shuttle must move between 2 levels. Initially, shuttle 4 is assigned to level 4 and 5. If the production capacity increases in the future, it might be needed to let the shuttles move between different levels, with as goal to provide enough supply for the production process. However, since this is not the main purpose of this research, we assume that the shuttles remain on their base level(s) only.

Input data
During the data collection phase, we came to the conclusion that not all the data was complete. In particular, the pallet height of a few stored products was missing. Company X uses a simple calculation tool to determine the height of pallets, by determining the number of layers there are on a single pallet and multiply that by the height of a single layer. In this way, the total height of the pallet is stored in the system. However, 6 of the 106 selected products did not have a registered height. Of these 6 products, 2 of them could be measured physically in the warehouse, whereas the other 4 products needed a different approach. The remaining 4 products were all products that could be compared to another product of which the height was available. For example, of product "9999", the height was available, while of product "9999B" the height was unknown. In the basis these two products are the same, but they have slightly different characteristics. Therefore, we used the height of the one product also for the comparable product.

Regarding the collected data, an additional assumption was made in the sense of the validity of it. The assumption was made that all the collected data is correct, with the underlying idea that the purpose of the research is not to validate the data, but to use it to determine the AS/R system performance. The data collection process was done in collaboration with warehouse operators and employees from the IT department, to ensure that the correct data is gathered.

Size of buffer area
In the employee availability section, we discussed that at the outbound station of the AS/R system, there is a buffer area that has a capacity of at most 5 pallets. This is not the only buffer area that exists, since there are buffer areas at each storage level as well. Before the elevator is able to move the pallets up and down, it must transport the pallets from the buffer area first. For example, when a pallet must be retrieved, the shuttle picks the specific pallet from the desired location, and drops it at the buffer area at the elevator at the corresponding storage level. Thereafter, the pallet is transported automatically without the use of a shuttle from the buffer area to the elevator, before it moves down. The assumption here is that the size of this buffer area is at most 1, with as underlying argument that the cycle amount is low enough to support this.

4.2 Optimization model and implementation

In order to optimize the storage strategy problem, we use a mathematical optimization model. Mathematical optimization, hereafter referred to as optimization, is a powerful tool to find the best possible solution in a set of alternative solutions. In this section, the following question is answered.

"How does the optimization model looks like and how do we implement the collected data?"

Any optimization model consists of an objective function, which is used to measure or assess the performance. In addition, sets include collections of related elements, parameters are fixed values that provide input for the model, and decision variables are the values that the model seeks to determine in order to optimize the objective function (Csanády et al., 2019). Our objective is to minimize the total travel time that is needed to complete all order requests. The relevant sets, parameters, decision variables, and objective function with constraints are presented below.

4.2.1 Rationale and motivation of the SLAP optimization model

Before we start with the actual formulation of the model, we discuss what requirements it should have in order to be an appropriate model. We have first consulted the literature, in which the research of Yang et al. (2013) deals with LASRSP for a multi-shuttle AS/R system. In our research we deal with a multi-shuttle AS/RS as well, but only consider SLAP instead of LASRSP. Because of this, the research of Yang et al. (2013) is a good starting point to formulate our model. Besides, the model by Yang et al. (2015) is a more simplified version of our model, and can therefore be used as well. However, we have changed the model in such a way, that it fits our purpose.

A major difference between the model of Yang et al. (2013) and this one, is that Yang et al. (2013) take into account both location assignment and storage/retrieval scheduling, while our model only takes into account the location assignment problem. If we compare only the storage location assignment problems (SLAP), Yang et al. (2013) do not really include any storage restrictions. As mentioned before, ingredients that must be stored have different heights, with as result that not all locations are possible for storage. Besides, ingredients that carry a specific allergen cannot be stored directly under or above an ingredient that carries another allergen. Products that are classified as non-allergens can only be stored above and not below products with allergens. Our model needs to take these hard constraints into account.

Third, the locations in the AS/R system considered by Yang et al. (2013) and Yang et al. (2015) are all directly accessible, while the locations in our AS/R system have more restrictions with respect to accessibility. In addition to that, Yang et al. (2013) use a Dynamic Programming approach, while our model can be classified as a Linear Programming model. On the other hand, the model of Yang et al. (2015) is an ILP model with operational cycles, which can be applied to our model as well. These points distinguish our model to the one of Yang et al. (2013) and Yang et al. (2015), at least for the SLAP case.

4.2.2 Validation of optimization model

With all constraints in mind we were able to formulate the complete SLAP optimization model. This model is shown in Appendix D – Complete optimization model. When we tried to evaluate the model with the complete problem instance, the program indicated that the problem was too large to solve exactly. A major limitation of this optimization model is the large number of constraints that we use. Allergen limits, heights, and capacity limitations are all hard constraints and therefore cannot be left out. As a result, the run time is very high even for a very small problem instance. We used the all the storage locations of the AS/RS, and with a run time of 5 hours and 15 minutes, we were able to store 180 pallets initially composed by 6 different types of ingredients. These 180 pallets are forming the initial state of the AS/RS at operational cycle $k = 0$. The retrieval or storage requests start at $k = 1$. The results are summarized in Table 4 below.

Request type	Operational cycle k	Ingredient	Location i	Location j	Location z	Objective (seconds)
Retrieval	1	A	9	28	1	55.55
Retrieval	2	B	9	26	3	77.94
Storage	3	B	43	28	3	77.94
Storage	4	A	43	28	1	55.55
Storage	5	C	41	26	3	77.94
Storage	6	D	41	26	1	55.55

Table 4: Results of SLAP optimization model with 7 cycles

When we increase the number of operational cycles to 8, the computational effort becomes too big to be handled by a regular computer. This is also dependent on the number of products that are stored initially, but with the current composition of only 2 products, 8 operational cycles becomes too large. In literature, several methods have been proposed to address situations like this one, and especially Roodbergen & Vis (2009) propose an extensive literature review on automated storage and retrieval systems, with as conclusion that the most common method to deal with large-scale problems is to use heuristics. However, before we continue to heuristics, we try to validate our SLAP optimization model first, to check whether it works as intended.

To show that the model works as intended, we create a dummy AS/RS with 2 levels, and each level having 3 by 3 storage positions. Storage locations 1 are always directly accessible, where behind locations 2 and eventually 3 are located. These are only directly accessible based on whether the location in front is occupied. The maximum storage height of level 1 is 1.5 meters and of level 2 is 2.0 meters. We have two different types of ingredients, that we call ingredient 1 and ingredient 2 for now. For both ingredients, 2 pallets must be stored initially. For both ingredients, 3 pallets must be stored before 2 pallets must be retrieved. Ingredient 1 is classified as a non-allergen, while ingredient 2 is carrying a milk allergen. The characteristics are summarized in Table 5 below.

Request type	Operational cycle k	Ingredient	Height (meters)	Allergen
Storage	0	1	1.6	Non
Storage	0	1	1.6	Non
Storage	0	2	1.3	Soy
Storage	0	2	1.3	Soy
Storage	1	1	1.6	Non
Storage	2	1	1.6	Non
Storage	3	1	1.6	Non
Storage	4	2	1.3	Soy
Storage	5	2	1.3	Soy
Storage	6	2	1.3	Soy
Retrieval	7	2	1.3	Soy
Retrieval	8	2	1.3	Soy
Retrieval	9	1	1.6	Non
Retrieval	10	1	1.6	Non

Table 5: Dummy problem instance with 2 ingredients

Figure 4 is a top-view of a 3 by 3 AS/RS where the main aisle is located at the south side, and the I/O is at the bottom left. The rectangles are storage locations, and included are numbers that indicate the ranking of travel time. The lower the number, the less time that is needed to reach this location. The lift time significantly influences the travel time for such a small problem instance, and therefore all the locations at level 1 can be reached before the best location at level 2 can be reached.

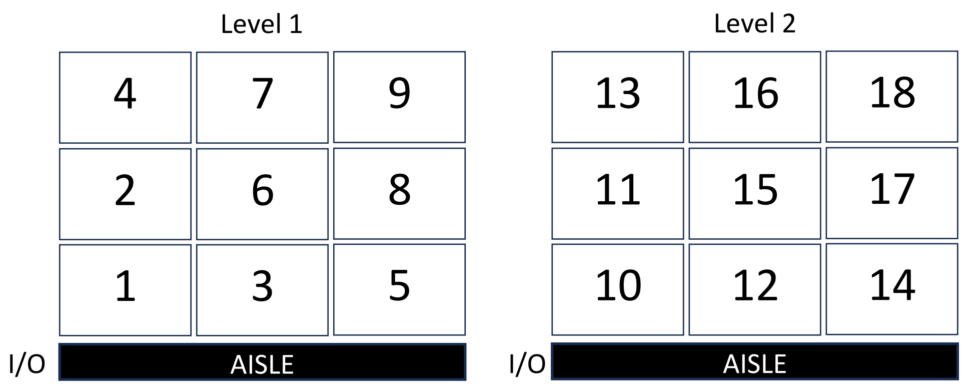

Figure 4: Ranking of dummy storage locations per level (1 for best, 18 for worst).

The outcome of the model should be clear, in the sense that ingredient 1 must be stored at level 2 and ingredient 1 at level 1. This is due to the maximum storage height limitation. The storage locations for ingredient 1 must be at locations (1,1,2), (1,2,2), and (2,1,2), indicated by rank 10, 11, and 12 respectively. It must be noted that location (1,2,2) must be visited before (1,1,2), otherwise it is not feasible. The same holds for ingredient 2, that must be stored at level 1 at locations (1,1,1), (1,2,1), and (2,1,1). These locations are indicated by rank 1, 2, and 3 respectively.

Request type	Operational cycle k	Ingredient	Location i	Location j	Location z	Objective (seconds)
Storage	0	1	1	3	2	-
Storage	0	1	2	2	2	-
Storage	0	2	2	3	1	-
Storage	0	2	3	2	2	-
Storage	1	1	2	1	2	78.28
Storage	2	1	1	2	2	78.02
Storage	3	1	1	1	2	75.27
Storage	4	2	2	1	1	58.56
Storage	5	2	1	2	1	58.30
Storage	6	2	1	1	1	55.55
Retrieval	7	2	1	1	1	55.55
Retrieval	8	2	1	2	1	58.30
Retrieval	9	1	1	1	2	75.27
Retrieval	10	1	1	2	2	78.02

Table 6: Results of the SLAP optimization model with the dummy instance

As we can see, the results are as expected. The total objective of this dummy instance equals 24.65 minutes. Ingredient 1 is stored constantly at level 2, while this is not necessarily needed for ingredient 2. It is not possible to store multiple ingredients in the same row, and that is also not happening. Another important thing to mention is that location (1,1,1) is retrieved before (1,2,1), and location (1,1,2) before (1,2,2). Besides for storage, location (1,2,2) is visited before location (1,1,2), and location (1,2,1) before (1,1,1). This indicates that the accessibility constraint is working as it should be, for the storage requests as well as for the retrieval requests. This indicates that the complete SLAP optimization model shown in Appendix D – Complete optimization model is working as it should and is thereby validated.

The complete problem instance that covers 13 weeks of operations has a total of 18,634 request cycles, of which 10,122 were in the form of storage, and the remaining 8,512 in the form of retrieval. The number of initial stored pallets equals 4,242, and comparing this with the small problem instance that needed 5 hours and 15 minutes for 180 initial pallets and 6 requests, this is significantly larger. The more pallets and requests we are able to use, the more representative the model becomes. This made us decide to leave out some constraints that are not classified as hard constraints, with as goal to achieve exact results with a somewhat larger problem instance. The reduced SLAP optimization model is shown below, represented by sets first, followed by parameters and decision variables, with the objective function and constraints thereafter.

4.2.3 SLAP optimization model formulation

Sets

I	Storage positions in length $i \in I$
J	Storage positions in width $j \in J$
Z	Storage level $z \in Z$
A	product allergen $a \in A$
P	Set of products $p \in P$
K	Set of all operation cycles in the planning horizon
RP_k	Products that must be retrieved $RP \subseteq P$ in cycle $k \in K$
SP_k	Products that must be stored $SP \subseteq P$ in cycle $k \in K$
L	Set of storage lanes $l \in L$
U	Unavailable storage locations $u \in U$

Parameters

h_p	height in millimeters of product $p \in P$
p_a	allergen $a \in A$ of product $p \in P$
h_z	maximum storage height of level z in millimeters $z \in Z$
$st_{i,j,z}$	time to travel from inbound to location i, j at level z, and back to the inbound
$rt_{i,j,z}$	time to travel from outbound to location i, j at level z, and back to the outbound
TT	total time for all requests in seconds
M	A large enough number
H_p^k	Number of pallets of product $p \in P$ stored during cycle $k \in K$

Variables

$$q_{i,j,z,p}^k = \begin{cases} 1 \text{ if location } i, j \text{ at level } z \text{ is occupied with product } p \text{ during cycle } k \\ 0 \text{ otherwise} \end{cases}$$

$$x_{i,j,z,p}^k = \begin{cases} 1 \text{ if product } p \text{ is retrieved from location } i, j \text{ at level } z \text{ during cycle } k \\ 0 \text{ otherwise} \end{cases}$$

$$d_{i,j,z,p}^k = \begin{cases} 1 \text{ if product } p \text{ is stored at location } i, j \text{ at level } z \text{ during cycle } k \\ 0 \text{ otherwise} \end{cases}$$

Objective Function

$$Min\ TT = \sum_{k=1}^{K}\left(\sum_{p=1}^{RP_k}\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{z=1}^{Z} rt_{i,j,z} * x_{i,j,z,p}^{k} + \sum_{p=1}^{SP_k}\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{z=1}^{Z} st_{i,j,z} * d_{i,j,z,p}^{k}\right)$$

Constraints

$$q_{i,j,z,p}^{k} * h_p \leq h_z \qquad \forall i, \forall j, \forall z, \forall p, \forall k \tag{1}$$

$$q_{i,j,z,p_a}^{k} + q_{i,j,z',p'_a}^{k} \leq 2 \qquad \forall i, \forall j, \forall k, \forall z' > z, a \in \{2,3\}, a' \in \{1\} \tag{2}$$

$$q_{i,j,z,p_a}^{k} + q_{i,j,z',p'_a}^{k} \leq 1 \qquad \forall i, \forall j, \forall k, \forall z' > z, a \in \{1,2,3\}, a' \in \{2,3\}, a \neq a' \tag{3}$$

$$q_{i,j,z,p}^{k} = 0 \qquad (i,j,z) \in U, \forall p, \forall k \tag{4}$$

$$\sum_{p=1}^{P} q_{i,j,z,p}^{k} \leq 1 \qquad \forall i, \forall j, \forall z, \forall k \tag{5}$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{z=1}^{Z} q_{i,j,z,p}^{k} = H_p^k \qquad \forall k, \forall p \tag{6}$$

$$x_{i,j,z,p}^{k} \leq q_{i,j,z,p}^{k} \qquad \forall i, \forall j, \forall z, \forall k, \forall p \in RP_k \tag{7}$$

$$M * \left(1 - d_{i,j,z,p'}^{k}\right) \geq \sum_{p=1}^{P} q_{i,j,z,p}^{k} \qquad \forall i, \forall j, \forall z, \forall p' \in SP_k, k \geq 1 \tag{8}$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{z=1}^{Z} x_{i,j,z,p}^{k} = 1 \qquad \forall k, \forall p \in RP_k \tag{9}$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{z=1}^{Z} d_{i,j,z,p}^{k} = 1 \qquad \forall k, \forall p \in SP_k \tag{10}$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{z=1}^{Z} q_{i,j,z,p}^{0} = P_p^0 \qquad \forall p \tag{11}$$

$$q_{i,j,z,p}^{k} = q_{i,j,z,p}^{k-1} + d_{i,j,z,p}^{k-1} - x_{i,j,z,p}^{k-1} \qquad \forall i, \forall j, \forall z, \forall p, \forall k \geq 1 \tag{12}$$

$$q_{i,j,z,p}^{k} \in \{0,1\} \qquad \forall i \in I, j \in J, z \in Z, p \in P, \forall k \in K \tag{13}$$

$$x_{i,j,z,p}^{k} \in \{0,1\} \qquad \forall i \in I, j \in J, z \in Z, p \in RP_k, \forall k \in K \tag{14}$$

$$d_{i,j,z,p}^{k} \in \{0,1\} \qquad \forall i \in I, j \in J, z \in Z, p \in SP_k, \forall k \in K \tag{15}$$

4.2.4 Explanation of constraints

In this section the constraints of the mathematical model are explained. These constraints influence the objective function, that minimizes the total travel time to handle all storage and retrieval requests. At each operational cycle k, a single retrieval or storage request is handled. In the objective function, the parameter x indicates the retrieval requests, and the parameter d indicates the storage requests. The first constraint ensures that if a pallet with product p is placed at location i, j at level z, the height of this pallet cannot exceed the maximum storage height of level z. Constraint (2) allows products classified as non-allergens to be placed above allergen-products. Constraint (3) only allows the same allergen types above each other, when $p_a{}'$ is not equal to 1 (Non-allergen). The exact rules are described in Section 2.2.1. Besides, for sake of simplicity we use sets I, J, and Z to indicate storage positions in length, width, and height respectively. However, main aisles, pillars, and elevators cannot be used for storage, and are therefore not classified as storage locations. All these unavailable locations are given by the set U, and constraint (4) guarantees that no product can be stored at these locations.

In addition to that, constraint (5) ensures that all locations cannot be occupied by more than one product at the same time. The reason for this is that at most a single product can be stored at a single pallet, and therefore this constraint ensures that not more than 1 type of product can be stored at a single location. Constraint (6) keeps track of the amount of pallets stored of a specific product during a specific cycle k, and constraint (7) allows a product to be picked from a location only if that product is actually stored at that specific location. Constraint (8) is making sure that the location is empty if a product is stored at that location. Constraint (9) and (10) ensure that all the retrieval requests and storage requests are fulfilled. Constraint (11) indicates that a starting stock level must be stored. Eventually, constraint (12) is the inventory balance equation, that ensures that the inventory level of a certain product is updated when a storage or retrieval request is performed. This is necessary when multiple operational cycles are considered. In our case, we start at $k = 0$ with a starting stock, and perform storage and retrieval requests from $k = 1$ onwards. Constraints (13), (14), and (15) are sign constraints, and can only take value 0 or 1.

4.2.5 Model input data

This section is an extension on Section 4.2.3, primarily to discuss the relevant data that is needed as input for the optimization model. Below, a summation is made with the used input data. Set I is used to represent storage positions along the length of the system, while set J is used for storage positions over the width. In **Error! Reference source not found.**, the values of set I start of the left hand side and e nd at the right hand side, while the values of set J start at the bottom and end at the top. Practically, this means that coordinate (i, j) = (1,1) is located at the bottom left hand side, and coordinate (i, j) = (54,38) is located at the top right corner. We also include the level z, resulting in a coordinate format of (i, j, z). Besides, for the allergen set A, we include non-allergens, milk allergens, and soy allergens. To simplify the use of this set, we use index 1 for non-allergens, 2 for milk allergens, and 3 for soy allergens.

Sets

Storage positions in length $I = \{1, 2, \dots, 54\}$

Storage positions in width $J = \{1, 2, \dots, 38\}$

Storage level $Z = \{1, 2, \dots, 5\}$

product allergen $A = \{1\ (Non), 2\ (Milk), 3\ (Soy)\}$

The next few sets are related to the ingredients that are stored and/or retrieved. The ingredients are also called products, and all have an unique name. However, for sake of simplicity we just use numbers. Initially, we had 83 unique products, but the model was able to evaluate exactly up to only 8 unique products. For the storage requests and retrieval requests, these are both sets with products from the original product set P, and are therefore both called multiset RP and SP. Again, for sake of simplicity we use numbers instead of product names. Again, we could not use the entire storage and retrieval instance, and therefore both sets are reduced to 70 storage requests, and 40 retrieval requests.

Stored products $P = \{1, 2, 3, 4, 5, 6, 7, 8\}$

Products that must be retrieved $RP = \{1, 2, \dots, 40\}$

Products that must be stored $SP = \{1, 2, \dots, 70\}$

Since only a single product can be stored or retrieved at a time, this would mean that we need to combine both RP and SP into a single set. This would give us the number of operation cycles K, but this brings a lot more complexity. In order to be able to evaluate the model exactly, K is reduced to a set of length of only 2.

Operation cycles $K = \{0, 1\}$

For the unavailable storage locations, denoted by U, a short explanation is desired. The size of set U equals 1,580, which refers to the number of unavailable storage locations. We can check whether this approach is correct, by multiplying the maximum number of I, J, and Z, and subtract 1,580 from that. The multiplication of these three numbers equal 10,260, and after subtracting 1,580 from that we end up with 8,680. This is exactly the maximum capacity of the AS/R system, and hence we have verified that the size of set U is correct. To present 1,580 values in an efficient way, we have created Table 7. In this table, 4 columns are presented with several rows containing sets. In order to find all unavailable coordinates, we take the Cartesian Product of i_n, j_n, and z_n for all n. The Cartesian Product or Unrestricted Join of sets A' and B' is denoted by $A' \times B'$ and results in a set of all ordered pairs (a', b') such that a' belongs to A' and b' to B' (Halpin & Morgan, 2008). In our case, we use 3 sets at a time for the Cartesian Product, and create coordinates with format (i, j, z).

Number n	**Length i**	**Width j**	**Level z**
1	$\{1, 2, \dots, 54\}$	$\{9, 18, 27\}$	$\{1, 2, 3, 4, 5\}$
2	$\{10, 42\}$	$\{10, 11, \dots, 26\}\backslash\{18\}$	$\{1, 2, 3, 4, 5\}$
3	$\{13, 27, 41\}$	$\{1, 2, \dots, 7\}, \{16, 25\}$	$\{1, 2, 3, 4, 5\}$
4	$\{27, 41\}$	$\{34, 35, \dots, 38\}$	$\{1, 2, 3, 4, 5\}$
5	$\{10, 11, \dots, 14\}$	$\{28, 29, \dots, 38\}$	$\{1, 3, 4\}$
6	$\{10, 11, \dots, 13\}$	$\{28, 29, \dots, 38\}$	$\{2, 5\}$
7	$\{14\}$	$\{32, 33, \dots, 38\}$	$\{2,5\}$
8	$\{42\}$	$\{28, 29, \dots, 38\}$	$\{1, 2, 3, 4, 5\}$
9	$\{43, 44\}$	$\{29, 30, \dots, 38\}$	$\{1\}$
10	$\{45\}$	$\{33, 34, \dots, 38\}$	$\{1\}$
11	$\{43, 44, 45\}$	$\{33, 34, \dots, 38\}$	$\{2, 3, 4, 5\}$
12	$\{13\}$	$\{26\}$	$\{1, 2, 3, 4, 5\}$

Table 7: Combination of sets representing unavailable storage locations

Parameters

Maximum storage height of level z in centimeters. $h_1 = h_2 = 150,\ h_3 = 180,\ h_4 = h_5 = 225$

Maximum pallet capacity of level z. $c_1 = 1{,}728,\ c_2 = c_5 = 1{,}740,\ c_3 = c_4 = 1{,}736$

4.2.6 Mutations made on optimization model

As said, the optimization model needs extremely much time to evaluate an exact solution, even for a very small problem instance. In order to have a more representative view of reality, a larger problem instance is desired.

Mixed storage lane constraint

The original model shown in Appendix D – Complete optimization model ensures that we cannot store more than 1 type of ingredient in storage lanes 1 and 4, and at most 2 different types of ingredients in storage lanes 2 and 3 at the same time. This is something that the stakeholders of Company X want to see, but it is not specifically a hard constraint. When leaving this constraint out, the outcome remains feasible, but according to the stakeholders at Company X, this is unlikely to happen in reality. Later in Chapter 5, we see what time is needed for relocating pallets compared to storing them separately with as result that all ingredients are directly accessible. The constraints we are referring to are shown directly below. The numbers at the end correspond to the constraint numbers in Appendix D – Complete optimization model.

$$\sum_{p=1}^{P} o_{i,l,z,p}^{k} \leq 1 \qquad \forall i, \forall z, \forall k, \forall l \in \{1,4\} \tag{14}$$

$$\sum_{p=1}^{P} o_{i,l,z,p}^{k} \leq 2 \qquad \forall i, \forall z, \forall k, \forall l \in \{2,3\} \tag{15}$$

$$\sum_{j \in J_l} q_{i,j,z,p}^{k} \leq M * o_{i,l,z,p}^{k} \qquad \forall i, \forall z, \forall l, \forall p, \forall k \tag{16}$$

$$\sum_{j \in J_l} q_{i,j,z,p}^{k} \geq o_{i,l,z,p}^{k} \qquad \forall i, \forall z, \forall l, \forall p, \forall k \tag{17}$$

If we look at the constraints above, we see that all constraints must hold for each operational cycle k. If we increase the number of storage and retrieval requests, and thereby the number of operational cycles, it becomes harder for the model to solve it exactly. Besides, when a higher number of operational cycles is used, more different types of products are used as well. Because the constraints sum over all products p, it becomes harder for the model to solve it exactly as well.

Accessibility constraint

The second set of constraints we omitted are the accessibility constraints. In a multi-deep AS/RS, one challenge is that not all storage locations are directly accessible, especially when the system is highly utilized. Ignoring this issue is unrealistic because, in reality, pallets that block certain locations must be relocated first, which can negatively affect the overall efficiency. However, we chose to leave this set of constraints out anyway, and the reason why can be found in the results that are obtained afterwards.

While initially, it might seem unrealistic to omit these accessibility constraints, our approach compresses multiple operational cycles into a single cycle. Although this does not perfectly reflect reality, it is a reasonable assumption under certain conditions. One of these conditions is that all storage and retrieval requests are handled at adjacent locations. In the original model, the system checks whether a location is accessible, but in this simplified approach, it only needs to verify if a location is empty (for storage) or occupied by the specific product (for retrieval). By processing requests at adjacent locations starting from a main aisle, we ensure that the handled requests remain feasible. The key assumption here is that we do not prioritize one request over another since we are using a single operational cycle. The specific constraints that were omitted correspond to those listed in the optimization model in Appendix D – Complete optimization model, and are shown directly below.

$$\sum_{p'=1}^{P} \sum_{j' \in RJ_l^1} q_{i,j',z,p}^{k} \leq M\left(1 - d_{i,j,z,p}^{k}\right) \qquad \forall i, \forall z, \forall p, \forall k, \forall l \in \{1,4\}\ \forall j \in J_l \tag{18}$$

$$\sum_{p'=1}^{P} \sum_{j' \in RJ_l^1} q_{i,j',z,p}^k \leq M\left(1 - x_{i,j,z,p}^k\right) \qquad \forall i, \forall z, \forall p, \forall k, \forall l \in \{1,4\}\ \forall j \in J_l \tag{19}$$

$$d_{i,j,z,p}^k \leq b_{i,j,z}^k \qquad \forall i, \forall j \in \{J_2, J_3\}, \forall z, \forall k, \forall p \in SP_k \tag{20}$$

$$x_{i,j,z,p}^k \leq b_{i,j,z}^k \qquad \forall i, \forall j \in \{J_2, J_3\}, \forall z, \forall k, \forall p \in RP_k \tag{21}$$

$$\sum_{p=1}^{P} \sum_{j' \in RJ_l^1} q_{i,j',z,p}^k \leq M * b_{i,j,z}^k \qquad \forall i, \forall z, \forall p, \forall k, \forall l \in \{2,3\}, \forall j \in J_l \tag{22}$$

$$\sum_{p=1}^{P} \sum_{j' \in RJ_l^2} q_{i,j',z,p}^k \leq M\left(1 - b_{i,j,z}^k\right) \qquad \forall i, \forall z, \forall p, \forall k, \forall l \in \{2,3\}, \forall j \in J_l \tag{23}$$

Similar to the mixed storage lane constraints, these constraints hold for all operational cycles, and 4 out of 6 constraints are also using a sum over all products that are present in the AS/R system. This means that when we increase the problem size, it becomes harder for the mathematical model to solve the problem exactly. With this approach, we hope to be able to increase the problem size significantly, at least to have a somewhat more better representation of reality. The results of the reduced SLAP optimization model are discussed in the next section.

4.2.7 Reduced SLAP optimization model outcomes

In the previous sections we discussed the smaller product instance that was used to be able to assess the optimization model. We started with an initial stock of 8 unique products, with a total number of occupied locations to be equal to 1,951. These 8 products were the ones that were moved most often during the past 12 months. Next to that, at $k = 1$, both storage requests and retrieval requests were added. At this operation cycle, 6 different products must be stored in a total quantity of 70 pallets. Next to that, 7 different products must be retrieved in a total quantity of 35 pallets. In the current model, we have incorporated that we do not take into account any storage time to store the initial stock at $k = 0$. The reason for this is that this is not a very everyday operation, as we consider a system that is replenished continuously rather than once in a while.

The optimal objective function for this set up equals approximately 1.99 hours. This time both includes storage and retrieval time, of which the latter is equal to 0.78 hours. This means that the total storage time equals 1.21 hours. All the locations for both request types are adjacent to each other, which means that for a single operation cycle, this is a feasible solution. When considering all the possible storage locations, and not only the lowest storage level as in Section 4.2.6, it is possible that some actions are performed at the same locations, but only at a different level. This is somewhat difficult to show, but 22 storage requests are handled at level 3, 2 are handled at level 4, and the remaining 46 are all handled at level 1. For the retrieval requests, 8 are handled at level 3, 5 at level 4, and the remaining 27 at level 1. In Figure 5, the initial storage map is shown without any further storage or retrieval requests. To highlight where the storage and retrieval requests take place exactly, we have created Figure 6. Here, we clearly see that the storage requests are handled near the inbound station, and the retrieval requests are handled near the outbound station.

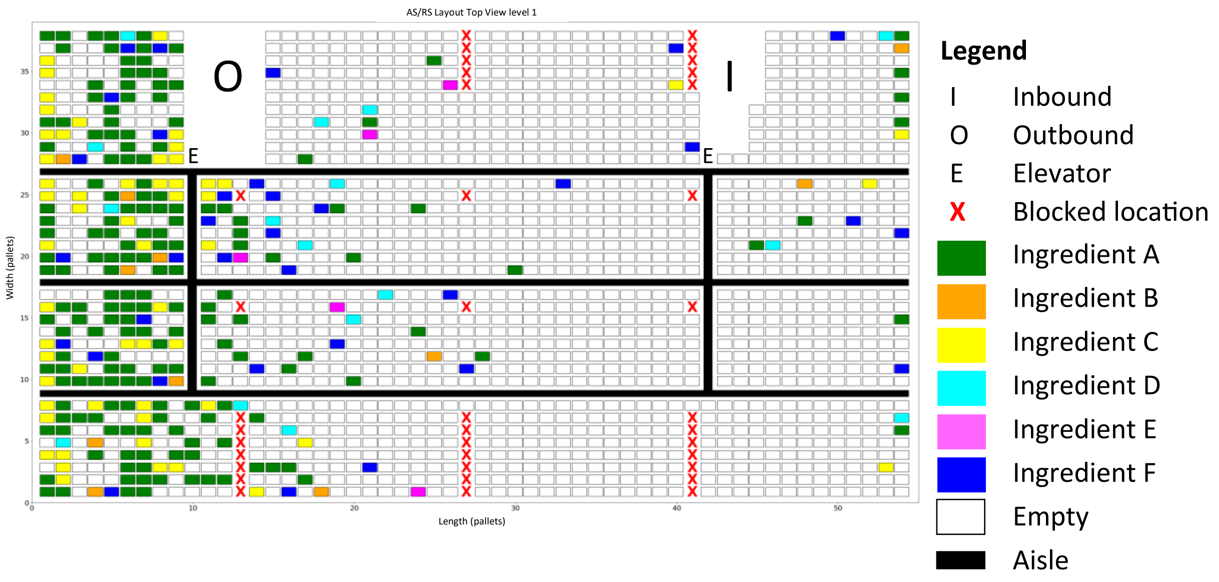

Figure 5: Storage map top view of level 1

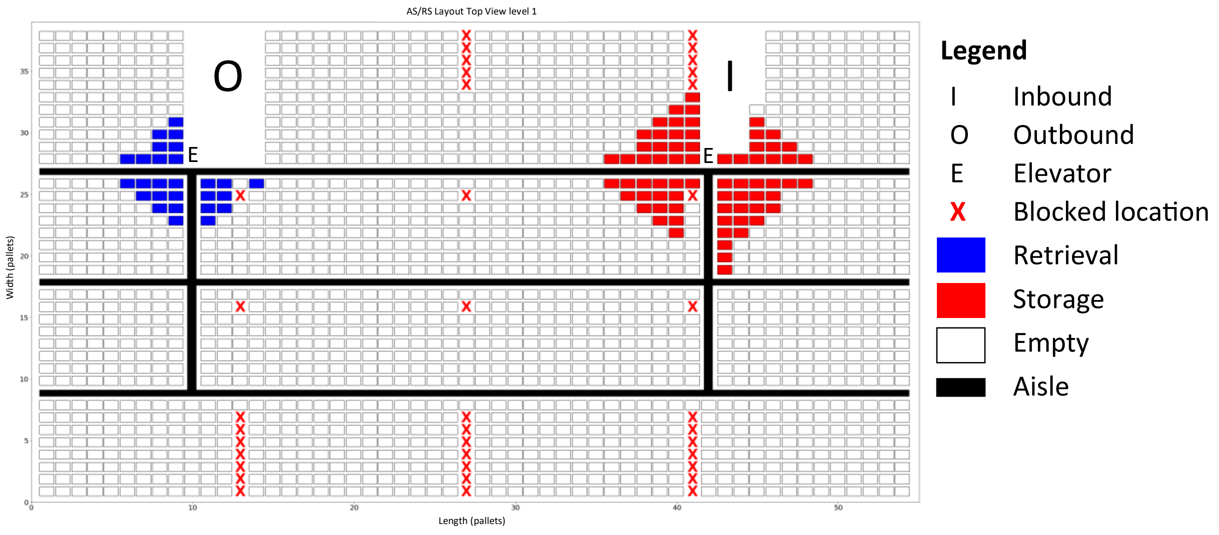

Figure 6: Indication of storage requests (red) and retrieval requests (blue)

4.3 Working of the simulation model

In this section, we describe the working of the simulation model that was established after we found out that the mathematical optimization model was not solvable exactly for the complete problem instance. The following question is answered in this section:

"What is the overall working of the simulation model?"

The first part of the simulation model imports data from several Excel files. These Excel files contain the most important information about the ingredients that are relevant, but also the characteristics such as the height of the pallet, the allergen of the ingredient, Identification, and batch size. On the other hand, we also have files that contain information about outgoing ingredient quantities, and incoming replenishment orders of ingredients. These latter two lists also contain dates and times. We have created the simulation model in such a way that it is able to handle both the complete problem instance and the small problem instance. Subsequently, we created a blank 3-dimensional area that represents the AS/RS. Here, we create 54 storage positions in length, 38 storage positions in width, and

5 storage levels. This, in combination with the unavailable storage locations creates exactly 8,680 storage positions for the AS/RS of Company X.

The majority of the simulation model is working the same for different storage strategies. The following points work the same for any storage policy. The simulation starts with initializing variables such as the number of pallets handled, the storage time and retrieval time, and the storage map. The storage map is initialized in order to create a feasible starting solution. This is necessary, since the 13-week period starts with retrieval requests. Besides, having an initial storage map before the request period begins is more representative than using an empty storage map.

Thereafter, the storage and retrieval requests are initialized and transformed in the correct format. Since the storage and retrieval requests both include date and time, we can create a single list that combines both request types and sort them from oldest to youngest. The simulation model handles these requests from oldest to youngest, and deletes the request from the list if it is complete. This enables us to check whether no request is skipped. The major difference between the format of the storage requests and the retrieval requests is that the retrieval requests are handled by one, and the storage requests are handled by replenishment batch size. This ensures that we are able to store the products based on batch number, as requested by Company X.

The simulation starts with first request, and thus handles the one with the oldest date. When the request is done, it moves to the next one until no requests are left. The model checks the type of request and based on that it determines where the shuttle should go. For example, when the next request is a retrieval request, it must send an empty shuttle, but if the next request is a storage request, the model must ensure that the shuttle is present at the inbound station first before the request can be handled. Based on the type of strategy, it determines where to store a pallet.

On the other hand, the retrieval of pallets is not dependent on the type of strategy, and always happens according to the same rules. This is due to the fact that the ingredients carry an expiry date, and to reduce waste, the ingredients with the fastest expiry date must be retrieved first. An assumption is made here, and that is that products that are stored earlier have a shorter expiry date than products that are stored later. This comes down on the FIFO principle. The simulation model searches for the ingredient that has the oldest storage date, and tries to retrieve this specific ingredient. If this is not possible, it checks the product with the second oldest storage date, and so on.

When a storage request consists of a large number of pallets, it could be possible that the next request already arrives when the current request is not done yet. If this next request is a retrieval request and the current request is a storage request, it pauses the current request to handle the next retrieval request first. This ensures that the simulation model also handles dual command cycles. The routing of the shuttles is dependent on the state of the storage map. This means that travelling from location A to location B does not always yield the same travel time. In a completely empty system, the shuttles will always travel the shortest route to the target location. The same yields for empty shuttles, because these can travel underneath the stored pallets. When a shuttle is carrying a pallet, it checks whether the shortest route is possible, otherwise it uses the main aisles and connection lanes.

The time a shuttle needs to travel a specific distance, and the way how this is calculated is the same for all storage strategies. After a single request is done, the travel time is calculated by keeping track of the travelled distance, in combination the handling operations the shuttle has performed. For example, if the shuttle has travelled a distance of 100 meters, made 4 turns, picked and dropped a pallet, we know exactly how much time was needed for this request. When a storage request comes in and the shuttle is not present at the inbound, the model inserts an additional request in which the travel time to the inbound is included for this request.

In the end, all the storage and retrieval requests are saved and an additional calculation enables us to see the total travel times of all shuttles per day. Besides that, we can see the difference between the total retrieval time and the total storage time in combination with the number of pallets handled. When all the simulation runs are completed, the simulation model transforms the collected results into an Excel file for visualization purposes. The general working of the simulation is constructed in pseudocode shown in Algorithm 1 below.

Algorithm 1 Pseudocode for the simulation model

```
for i = 1 to num_simulations do:
    initialize storage_time, retrieval_time, pallets_handled, requests_handled to 0
    initialize rejected_requests as empty lists
    initialize total_travel_time_per_day as empty dictionary
    clear the storage map
    create empty DataFrame with columns LocationX, LocationY, LocationZ, and TotalTime
    if improvement phase is equal to False then:
        if strategy is "Dedicated" then:
            initialize dedicated storage map
        else:
            initialize normal storage map
    else:
        copy candidate_map to storage_map
        if improvement_strategy is "Normal" or "1full" or "2full" or "Partial" then:
            propose swap with selected improvement_strategy
            copy storage_map to store_map_after_swap
    initialize storage_list and retrieval_list
    create combined_list from storage_list and retrieval_list and sort it based on date and time
    while combined_list is not empty do:
        initialize stored_amount to 0
        pop the first element from combined_list as current_request
        if combined_list is not empty then:
            set next_request to the first element in combined_list
        else
            set next_requets to None
        evaluate current_request
        if current_request has no valid coordinates then:
            append current_request to rejected_requests
        if request_amount is valid then:
            assign shuttle to handle the request
            determine travel path and calculate total travel time
            update shuttle start times and end times, and track total travel time per day
            if request_type is "Storage" then:
                update storage map with new storage coordinates
                increment counters
            else:
                clear storage map entry for retrieval
                increment counters
            check next_request and adjust if arrival time < current time
    calculate total pallet amount
    generate new results from simulation
return final results in Excel file
```

4.4 Storage policies used

In this section we elaborate on the chosen storage strategies and their results. The following question is answered:

"What types of storage policies do we use in our model to determine the performance of the AS/RS?"

Before we continue, we have described several storage policies in Chapter 3, each with their own advantages and disadvantages. Therefore, it would be nice to see which storage policy performs most closely, if not the same, to the optimal solution found by the mathematical optimization model. If this is done, we can test the complete problem instance as well, since a simulation model is a suitable method to deal with a larger problem instance like this one. This enables us to assess the performance of the AS/R system on the complete problem instance as well, and not only the smaller instance used for the optimization model. The most common storage strategies applied in literature are random, class-based, and dedicated. For the random and class-based strategies, randomness is involved. This means that we have to be careful with drawing conclusions directly after the first run. Statistically, it is possible that a single outcome turns out extraordinary high or low, and therefore does not represent the actual performance well. In the next section, the number of replications is discussed. In the sections thereafter, the 3 storage policies are discussed. Within these sections, we start with the small problem instance that corresponds to the one used in the mathematical optimization model, and continue with the complete problem instance.

4.4.1 Number of replications

The results are dependent on multiple factors, but one is of major importance when doing a simulation study. The number of replications confirms the reproducibility of the simulation, and refers to the independent verification of prior findings. Moreover, replication refers to the ability to independently replicate and reproduce computations (Arifin & Madey, 2019). Therefore, the number of replications directly affects the quality of the results. The higher number of replications is, the more accurate the results are. A higher number of replications brings as disadvantage that a longer runtime is required. To determine the least amount of replications with the highest possible accuracy of results, we use the confidence interval approach. The goal of the confidence interval approach is to find a number of replications after which the confidence interval (CI) becomes sufficiently small. To calculate the confidence interval, Equation (11) is used.

$$CI = \left(\bar{X} - z_{1-\frac{\alpha}{2}}\frac{\sigma}{\sqrt{n}}; \bar{X} + z_{1-\frac{\alpha}{2}}\frac{\sigma}{\sqrt{n}}\right) \tag{11}$$

Where $\bar{X}$ is the moving average over the number of replications, α is the level of significance, σ is the moving standard deviation, and n is the number of replications. A value for the level of significance that is often used is 0.05, and using this α means having a Z-score of $z_{0.975}$, which equals 1.96. In order to determine when the confidence interval is sufficiently small, we initialize the number of replications to 50. We expect to see the confidence interval width decrease, and at some point, the rate of decrease will become so low that it is not worth the effort to use additional replications for a minor reduction in interval width. The results of the interval width calculations are shown in Figure 7 below.

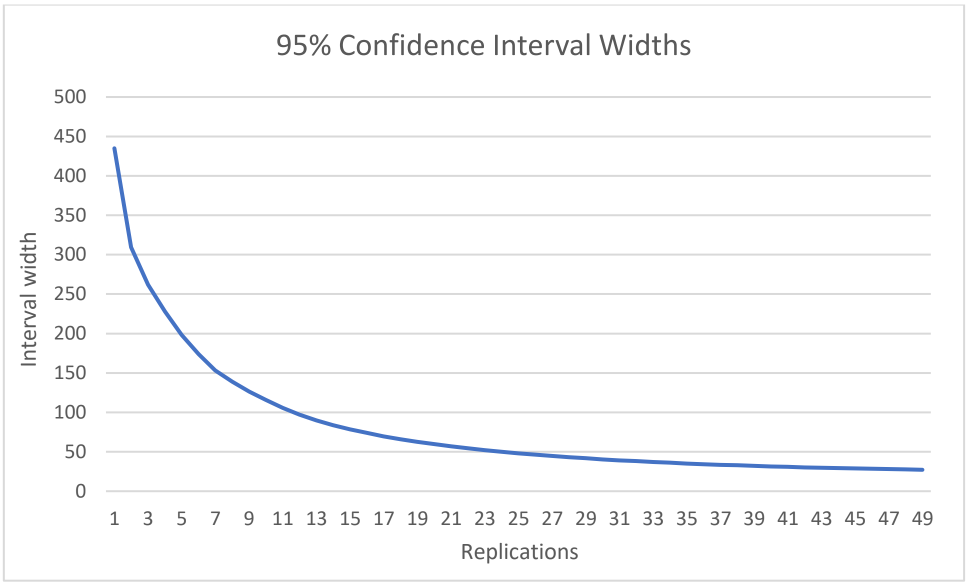

Figure 7: Confidence Interval widths based on the number of replications

Before the data was analyzed, we converted the end results from seconds to minutes. The graph above clearly visualizes that the rate of decrease is much higher in the beginning, while it becomes much lower towards the end. This means that the outcome is in line with the expectations we sketched before. The graph above does not show a very fast decay, with as result that a higher number of replications is needed to get accurate results. However, if we look at the data and calculate the relative change per replication, we see that after 20 replications the relative change becomes less than 5% for the first time. Therefore, we set the number of replications equal to 20.

4.4.2 Random strategy

The first storage policy that was used is the random storage policy, and both the small problem instance as well as the complete problem instance are analyzed.

Small instance

We began by analyzing the small problem instance to compare it with the optimal solution provided by the mathematical optimization model. This resulted in a total travel time of 3.73 hours. For the retrieval requests, on average 1.3 hours were needed, while for the storage requests 2.42 hours were needed. This means that on average, 2.03 minutes were needed to handle a single pallet. If we compare this to the optimal solution found, this is roughly 87% higher. Of course, with a small problem instance there is a lot more room for improvement, but this indicates that a random storage solution could lead to a far from optimal solution.

Complete instance

Since we are using a simulation model, we are able to assess the performance of the AS/R system on a complete instance as well. The first trial was based on the product set described in Section 2.3.3, but we came rapidly to a conclusion that this was not feasible as well. We started with the products that have the highest movement occurrences, and continued until all the 106 products were stored. However, it was not possible to store all the products, mainly due to the restrictions with respect to storage height and allergens. This enforced us to use a different approach again.

In accordance with the stakeholders at Company X, the product list in Chapter 2 was created to represent the future stock level as good as possible. The total stock amount of this list equals 7,764 pallets, which could not be stored completely with the current heights and allergens configuration. Next to that, the pallet movements were analyzed as well, and resulted in a list that covered a time span of 13 weeks. The starting date of this list was the first of January 2024, and ended at the last day of March 2024. This made us decide to look at the stock levels of the first of January 2024, with as goal to reduce the size of the product list. This list might not be averages or the best representative for an entire year, but this enables us to retrieve what is needed, since we are looking at an order list that starts at January 1. We also incorporate a replenishment list over the same time span, to maintain stock levels. At the first of January 2024, there were 83 different types of ingredients on stock, while the initial data had 106. The total number of pallets on stock at the first day of 2024 was 4,242. This should enable us to create a feasible initial solution.

The allergen and height restrictions again caused an unwanted outcome. When assigning products completely random to the AS/R system, approximately 250 pallets were not stored on a constant base. The reason for this was simple, namely that the approach did not take into account any logic when placing pallets. A simple example illustrates this. A product classified as a non-allergen forbids placing any allergen-carrying products above it. When level 1 of the AS/R system is completely filled with non-allergen products, not a single allergen-carrying product can be placed in the system anymore. Even if the system is for approximately 20% occupied, this situation results in an initial solution that is only partly feasible.

In order to create a solution that is completely feasible, we incorporated a small storage assignment logic. This small logic ensures that the emphasis of placing non-allergen products lies at the higher levels, and for the allergen-carrying product at the lower levels. Moreover, this does not mean that allergen-carrying products cannot be placed at the highest levels, but it focuses on placing these types of products on lower levels. Eventually, this resulted in an initial solution where all the 4,242 pallets were placed.

The small storage assignment logic ensured that we first check the higher levels for the non-allergen products, and first the lower levels for the allergen products. Specifically, this includes that for non-allergen products we select a random lane at level 4 first, and check if it is feasible. If not, we continue to level 5, and eventually to level 3, level 2, and level 1. With this approach, it is more likely that non-allergen products are placed at higher levels. Allergen-carrying products started a random search at level 1, and then continue sequentially to levels 2, 3, 4, and 5. Considering the initial random solution and all operations over a time span of 13 weeks and running 20 simulations, the average total time that was needed to process all the requests equals 582.27 hours. For the all the retrieval requests, 294.54 hours were needed, while for all the storage requests 287.73 hours were needed. These outcomes are based on the simulation model that takes into account the possibility of having dual command cycles.

4.4.3 Class-based strategy

The second storage strategy that was used is the class-based strategy. We apply an ABC-strategy where we both assign locations and products to class A, B, or C. The classes given to the locations are not based on single locations, but on the entire storage lane. The reason for this is that we prefer to store a single item in a single lane, especially the lanes that are accessible from one side only. It would therefore not be convenient to assign class A to locations at the beginning of the lane, while the locations at the end of the lane have class B. Besides, the product classes are determined based on the movement count. The number of times an ingredient is needed for production is counted and sorted, and the cumulative contribution per ingredient is calculated. The first 75% of the movements belong to class A, which is achieved by 23 different products. The next 20%, thus covering 95% of the movements in total, belong to class B. The last 5 cumulative percentages are assigned to class C. The class B movements are achieved by 23 different products as well, while 69 different products belong to class C.

Small instance

The small problem instance included only 8 unique products, and since we do not use a real retrieval list, it is hard to determine the actual class of the products. In addition to that, with a total capacity of 8,680 pallets, and only 1,951 of these locations are used, assigning classes A, B, or C to the closest 1,951 locations would not make a major difference. Last, since we are using single command cycles only for the small instance, the locations classified as A, B, and C can be different than for dual command cycles. For retrieval requests, A locations would lie more towards the top left corner of the storage map, while for storage requests A locations lie more at the top right corner. To conclude, these points made us decide to classify all 8 products to class A.

Subsequently, this leaves us with the question which locations to classify as class A locations. Since we consider single command cycles, for both storage and retrieval operations, no single location is better than another for all operations. For sake of simplicity, we use storage lanes 3 and 4 only, as indicated by the green marked lanes in **Error! Reference source not found.**. Again, 20 replications are used since r andomness is involved, and the average total travel time ended up to equal to 2.99 hours. Of this total objective, 1.01 hours were spent on retrieving products, while the other 1.98 hours were spent on storing products.

Complete instance

The complete problem instance covered 83 unique products with a total stock level of 4,242 pallets. With this in mind, we decided to enable the entire AS/RS, rather than only storage lanes 3 and 4. Since the complete instance is able to handle dual command cycles, we cannot use the same strategy for assigning classes to ingredients and locations as we did for the small instance. To determine the most preferred locations, we started from the inbound station, traveled to each location, and traveled then to the outbound. The result of this is that we are now able to distinguish storage positions that take more time to reach than others. To visualize this, **Error! Reference source not found.** is created. In this f igure, the locations that need less time to reach are indicated by red colors, and evolve to yellow, green, blue, and purple for locations that need more and more time to reach. This helped us to identify the storage lanes for class A, B, and C. In **Error! Reference source not found.**, the red locations are classified as class A locations, the green locations as class B, and the purple locations as class C.

The incoming ingredients need to have a classification as well, since we are then able to assign a certain location to it. The entire order list of the first 13 weeks of 2024 was checked and the number of retrievals per product was counted. The list was sorted in descending order based on retrieval frequency. In total, 20 unique products contributed to more than 75% of all the retrievals. These products are classified as A products. The next 21 products contributed approximately 18% of the total retrievals, and were classified as B products. The last 33 products contributed for less than 4% to the total retrievals, and are therefore classified as C products.

The next step is to create an initial solution again with the proposed class-based strategy. This was done according to the same small logic proposed for the initial random solution, where allergen-carrying products were checked for the lower levels first, while the non-allergen products were checked for the higher levels first. When a product needs to be placed, we check the class type of the corresponding product and randomly select a storage lane within the relevant class. When no lane is available at a certain level, we move up or down dependent on the allergen characteristics, and try to select another random storage lane within the class. This resulted in an entire feasible initial solution where 4,242 pallets were stored. The same replenishment list and order list in combination with the request sequence handling logic was used and this resulted in an average total travel time of 504.46 hours over 20 simulation runs, of which 249.24 hours were assigned to retrieval requests, and 255.22 hours to storage requests.

4.4.4 Dedicated strategy

The third implemented strategy is the dedicated policy. Previously, we mentioned that storing almost 8000 pallets is not feasible with the current product portfolio. A major disadvantage of the dedicated strategy is that it needs a number of storage locations equal to the maximum inventory per ingredient over the relevant period. In our case, this period ranges from the first of January 2024 until the end of March 2024. In the previous sections, two different storage policies were discussed that both incorporate randomness to obtain a solution. However, for the dedicated strategy this is not the case, since predetermined locations are used for storage and retrieval. Besides, no randomness is involved in the storage and retrieval list, and therefore we only need a single simulation run instead of 20.

Small instance

For the small instance, we checked the maximum stock level per product and used that to predetermine the locations. To find these maximum stock levels, we added the 6 storage requests to the corresponding stock level, and ended up with a total number of pallets of 2,021. Thereafter, we stored the initial stock levels and assessed the performance based on the storage and retrieval requests. For the latter, 1.09 hours were needed to retrieve 40 pallets. The other 1.72 hours were needed for storage requests. Thus, the total time needed to handle all requests equals 2.81 hours.

Complete instance

The complete instance requires more attention for a successful implementation. The main reason for this is that the maximum stock levels over a period of 13 weeks fluctuate a lot more compared to the small instance. Where initially at the first of January 2024 only 4,242 pallets needed to be stored, adding up the maximum stock levels of all the products over the 13-week period resulted in a total stock level of 7,820 pallets. When we tried to store the product list established in Chapter 2 before, this resulted in an infeasible solution.

Even when using the small storage assignment logic, where the emphasis of storing allergen products lies at the lower levels, and for storing non-allergen products at the higher levels, this resulted in an infeasible solution. In addition, we have incorporated an additional assignment logic that starts with assigning products that are moved most frequently to the locations that need the least travel time to reach. Eventually, this did not result in a feasible solution as well. At this point, we have implemented different assignment logics with as goal to create a feasible solution, however without a success. Therefore, we introduce a different method that might help.

We started again with an empty map and started to reserve the locations for each product. The two assignment logics from the paragraph above were used for this, in combination with a function that keeps track of the number of locations reserved for each product. When this number of reserved locations is less than the predetermined maximum storage amount, this means that there are no storage lanes available anymore for a feasible storage solution. When this happens, we enable a swap function, that swaps two random storage lanes and then checks whether a lane becomes available.

The swap function works as follows. We randomly select a single storage lane and then try to swap it with a randomly selected other storage lane. It is possible to use different swaps, for example, it could be the case that we swap two fully occupied lanes with each other, an empty lane with a fully occupied lane, or an empty lane with a partial occupied lane. When two lanes are selected for a potential swap, we first check whether restrictions exists that make the swap infeasible. If not, the swap is performed and the storage map is updated and checked on potential new storage positions afterwards. When the swap is not feasible, it can select up to 500 times a new storage lane with the original one. If after 500 trials there is still no swap available, we randomly select another storage lane and check potentially up to 500 new storage lanes for swaps. This process can take up to 500 times, which means that potentially

500^2 swaps are possible. This resulted in a feasible solution, which meant that we can also assess the incoming storage and retrieval requests. This resulted in a total objective value of 673.13 hours.

4.5 Conclusion

To end this chapter we provide a concise conclusion about the findings and results from this chapter. It was clear from the beginning that a set of assumptions was desired, especially when we found out that the mathematical optimization model for SLAP could not solve the complete problem instance exactly. We have confirmed that the complete optimization model works, whereafter we decided to omit 2 sets of constraints. This enabled us to use a larger problem instance, and thereby providing more reliable results. The SLAP optimization model without the 2 sets of constraints resulted in an optimal solution of 1.99 hours to handle all 110 pallets. Thereafter, we have assessed the performance of the simulation model with 3 types of storage strategies, namely, random, class-based, and dedicated. The class-based storage strategy performed the best here, with a total time of 3.89 hours to perform all requests. This is significantly higher than the optimal solution found by the optimization model, and therefore we continue with an improvement phase in the next chapter.

5 Improvement phase and analysis of results

In this chapter, we evaluate the performance of the used storage strategies from the previous chapter, and potentially improve the results. This is all done in Section 5.1. Thereafter, we perform an utilization analysis in Section 5.2. Last in Section 5.3, we analyze the results to be able to determine whether the improvement phase actually worked or not.

5.1 Improvement phase

In this section, we shortly look back at the results found so far, and try to improve the solutions in such a way that it benefits the efficiency of the AS/R system. The purpose of using the mathematical optimization model with a small instance is to determine the optimal performance based on this small instance. Subsequently, the simulation model can be used to evaluate the entire problem instance, since it uses a heuristic to create a solution. Solutions that are created heuristically are not necessarily optimal, and therefore it is important to check how good the solution actually is.

“How do we improve the results found so far and which methods can be used for that?”

We are able to determine the performance of the heuristic solutions, since we did not only use the simulation model on the complete problem instance, but also on the small instance. This enables us to check the performance of the heuristic solution compared to the optimal solution, and thereby also draw a careful conclusion about how good the heuristic approach performs on the complete problem instance. To make this conclusion more precise, we can try to improve our heuristic solution by using a metaheuristic approach. Genetic Algorithms usually need more computational effort to reach near optimal or even optimal solutions. Hence, Simulated Annealing and Variable Neighborhood Search are more suitable options. Since we are working with a minimization problem, we aim to find an objective value that is as small as possible.

Simulated Annealing

Simulated Annealing, hereafter referred to as SA, is a probabilistic metaheuristic technique that uses this to be able to escape local optima. But before SA uses this probabilistic approach, it first generates an initial solution with a starting temperature. From this initial solution, a perturbation is made to create a neighborhood solution. If this neighborhood solution is better, it is always accepted. If it is worse, it is accepted based on a probability that decreases as the quality of the neighborhood solution declines and as the temperature lowers. This temperature gradually decreases based on a cooling constant (α), with as goal to reduce the probability of accepting worse solutions over time. We have implemented this approach, and this is presented by the pseudocode in Algorithm 2 (Leeftink, 2022):

Algorithm 2 Simulated Annealing

```
1 Temperature = StartingTemperature
2 Solution = CreateInitialSolution
3 CurrentBest = 9,999,999
4 while not stopping_criteria do
5       for i = 1 to NumberOfIterations do
6             NeighborSolution = CreateNeighborSolution(Solution)
7             if NeighborSolution < Solution then
8                   if NeighborSolution < CurrentBest then
9                         CurrentBest = NeighborSolution
10                  end if
11                  Solution = NeighborSolution
12            else
13                  if RandomNumber ≤ $exp\left(\frac{CurrentSolution - NeighborSolution}{Temperature}\right)$ then
14                        Solution = NeighborSolution
15                  end if
16            end if
17      end for
18      Temperature = $\alpha$ * Temperature
19 end while
20 Result = CurrentBest
```

For the SA approach, the quality of the end solution is not only dependent on the initial solution, but also on the chosen parameters. The decisions must be made for the number of iterations, the starting temperature, and the cooling constant. We have tested several combinations of parameters, of which the results are shown in Table 8 below. To increase the computational efficiency, we used the small problem instance instead of the complete problem instance. In addition, the class-based storage strategy performed best for the small problem instance, what made us decide to choose this strategy as starting solution. Besides, the results are related to the retrieval times only, and thus are not influenced by storage requests. The initial solution and best solution values are both shown in minutes.

Iterations	Starting Temperature	Cooling Constant	Initial Solution	Best Solution	Improvement
1000	10	0.99	56.17	52.07	7.30%
1000	100	0.99	56.60	52.57	7.12%
1000	200	0.99	57.01	50.12	12.09%
500	10	0.99	54.73	51.84	5.28%
500	100	0.99	54.92	50.25	8.50%
500	200	0.99	57.36	50.42	12.10%
1000	10	0.995	56.49	52.99	6.20%
1000	100	0.995	54.32	50.11	7.75%
1000	200	0.995	57.79	50.11	13.29%
500	10	0.995	56.81	51.49	9.36%
500	100	0.995	54.95	51.31	6.62%
500	200	0.995	60.68	50.50	16.78%

Table 8: Results of the simulated annealing parameter analysis

From the results above, we clearly see that a higher starting temperature ensures that we more likely end up with a better solution. The highest improvement is achieved by 500 iterations in combination with a starting temperature of 200 and a cooling constant of 0.995. However, this configuration does not lead to the best solution found so far. The best found solution equals 50.11 minutes in retrieval time, which equals 1.25 minutes per request on average. This solution is achieved by using 1000 iterations and a cooling constant of 0.995, in combination with a starting temperature of 100 or 200.

The results from Table 8 above are obtained by using the class-based storage strategy with 3 classes. We also tested the random storage strategy with 1000 iterations, a starting temperature of 200, and a cooling constant of 0.995. The initial solution for this configuration equals 4.15 hours, of which 1.45 hours were spent on retrieval requests. After the SA algorithm improved the solution, we ended up with a solution that equals 2.95 hours. The total travel time is reduced from 1.45 to 1.10 hours, which is equal to a decrease of 23.87%. The solution found for the retrieval time is significantly higher than the solution found by using the class-based strategy. This implies that the solution is dependent on the quality of the initial solution, despite the fact that solution is improved by 23.87%.

The optimal objective found by the optimization model equals 1.99 hours, of which 0.78 hours were spent on retrieval requests. If we compare that with the solutions found after using the SA algorithm, we observe an optimality gap of 41.45% for the random strategy, and for the class-based strategy a gap of 7.19%. An important remark has to be made here, and that is that in the simulation model we stick to the preference of Company X to store at most a single lot number in storage lanes 1 and 4 and at most 2 lot numbers in storage lanes 2 and 3. Comparing this to the optimal value from the optimization model, we see that in the optimization model more than a single type of product is stored in storage lane 4. This most likely is the cause of still having a small optimality gap after the SA algorithm was used.

Additionally, the time it takes to retrieve a product is dependent on the current state of the system. Especially locations in storage lanes 2 and 3, i.e. the lanes that are accessible by 2 sides, could have different travel times when the lane is empty or not. In the mathematical optimization model, we used the shortest possible path to the specific location for both storage and retrieval requests. In the simulation model, the shuttle determines the shortest possible path, by taking into account the current occupied locations. Dependent on whether the shuttle is empty or not, it determines the shortest possible path that is possible to reach the target location.

The next analysis is done on the complete problem instance, where 4,242 pallets were stored initially, 8,512 pallets must be retrieved, and 10,122 pallets must be stored. We used the parameter configuration that was determined earlier, with 1000 iterations, a starting temperature of 200, and a cooling constant equal to 0.995. The initial retrieval time equals 248.28 hours, and after improvement the total retrieval time needed was 243.54 hours. The improvement made is only equal to 1.95%, but there is a likely reason for this.

First of all, the basic idea behind retrieving products in the warehouse of Company X is that the products with the shortest expiry date are retrieved first, in order to prevent expired ingredients. By using an initial stock level of 4,242 pallets, and a 13-week order period in which 8,512 pallets must be retrieved, and 10,122 pallets must be stored, this results in that at least 4,270 (8,512 minus 4,242) pallets that must be retrieved were on the storage list as well. Therefore, the retrieval objective is not only influenced by the way how products are located in the initial state, but also on the way how products are stored. When the retrieval objective is only dependent on the initial state, for example for the small problem instance, the improvements by the SA algorithm are potentially higher.

Variable Neighborhood Search

The second metaheuristic approach that was used is Variable Neighborhood Search, in short VNS. The first step is to initialize a starting solution, in combination with determining the different neighborhood structures that are going to be used. From this starting solution, a neighborhood solution is created by using a specific neighborhood structure. A local search method is used to find the local optimum in the current neighborhood solution. If an improvement is found, the solution is updated and the neighborhood structure is reset to the first one. If no improvement is found, the neighborhood structure is incremented until it exceeds the maximum number of neighborhood structures. The steps of the VNS algorithm are shown in Algorithm 3 below:

Algorithm 3 Variable Neighborhood Search

```
1 Solution = CreateInitialSolution
2 CurrentBest = 9,999,999
3 Neighborhoods = [Type I, Type II, Type III]
4 for i = 1 to NumberOfIterations do
5       k = 1
6       while k ≤ k_max do
7             Strategy = Neighborhoods(k)
8             Solution = LocalSearch(Strategy)
9             if Solution ≤ CurrentBest then
10                  CurrentBest = Solution
11                  k = 1
12            else
13                  k = k + 1
14            end if
15      end while
16 end for
17 Result = CurrentBest
```

As discussed before, the VNS algorithm uses a local search method for exploring the current neighborhood. The process of exploring the current neighborhood solution is referred to as intensification, while creating different neighborhoods is referred to as diversification. For clarity, the pseudocode of the local search method is given by Algorithm 4 below.

Algorithm 4 Local Search

```
1 NumberOfIterations = 100
2 NoImprovementLimit = 50
3 NoImprovementCount = 0
4 BestSolution = 9,999,999
5 while not NoImprovementCount ≥ NoImprovementLimit do
6       for i = 1 to NumberOfIterations do
7             Solution = CreateNeighborhoodSolution(Strategy)
8             if Solution ≤ BestSolution then
9                   BestSolution = Solution
10                  NoImprovementCount = 0
11            else
12                  NoImprovementCount = NoImprovementCount + 1
13            end if
14      end for
15 end while
16 Result = BestSolution
```

A common neighborhood structure for the VNS algorithm when dealing with an integrated storage and retrieval problem, is to intercept or interchange storage and or retrieval requests in existing request cycles. This is incidentally applied by Yang et al. (2015) as well. For our research, this approach is less useful. In the first place, only single command cycles in combination with a single operational cycle are used. This means that we cannot change storage and or retrieval requests, since these are all handled separately in a single cycle. Second, the production planning department of Company X sends orders to the warehouse with required ingredient quantities for production. A request that comes in at the beginning of the day cannot be interchanged with an order at the end of the day, because that messes up the production schedule. Third, incoming truck loads that replenish the stock quantities must be handled according the first-come-first-serve (FCFS) principle, and can therefore not be interchanged without consequences. Therefore, this neighborhood structure for the VNS drops as suitable option.

An additional method that is commonly used in VNS to explore the solution space is repair and destroy. As the name suggests, the destroy method destructs a part of the initial solution, and thereby enabling the repair method to rebuild the destroyed part. The destroy method typically contains some form of stochasticity, such that not the same parts are destroyed in every invocation of the method (Pisinger & Ropke, 2010). In our research, destroy and repair methods are not used, but rather different types of swaps as discussed above.

Yang et al. (2015) propose an optimization model for a storage/retrieval problem, in combination with a VNS heuristic for SLAP. We know that the quality of the initial solution affects the quality of the end solution. Therefore, we select the solution obtained with the class-based storage strategy as the initial solution. The next step is to determine different neighborhoods, for which we selected 3 types. Two of these neighborhoods are to some extent applied by Yang et al. (2015) as well, and these are swapping an empty storage lane with an fully occupied storage lane, and second swapping two fully occupied storage lanes. The third neighborhood is created by swapping a part of an occupied storage lane with a part of another occupied storage lane. This principle is shown in Figure 8 below, where 2 storage lanes next to each other are shown from above. The colors represent a type of product, and the white cells indicate an empty storage location. Obviously, finding a neighborhood is not restricted to two storage lanes that are located next to each other.

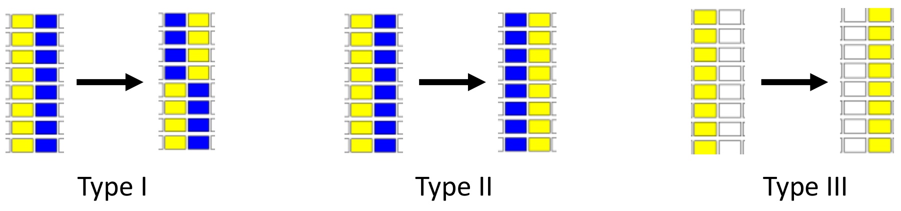

Figure 8: Visualization of neighborhood structures

For the small instance, we tested the VNS metaheuristic with the 3 neighborhood structures described above. We set the number of iterations equal to 5, and the local search iterations equal to 100. The first results were obtained for the small problem instance by using the class-based storage strategy. The result of this approach is that we ended up with a minimum total travel time that was needed for all requests of 2.66 hours. 0.9 hours of this total objective was needed to handle the retrieval requests, and the remaining 1.76 hours were spent on storage requests. Besides, we tested the random storage strategy as well, and after using the VNS algorithm, the total travel time that was needed equals 3.12 hours. Of this total objective, 1.13 hours were spent on retrieving products, and the other 1.99 hours were used for handling storage requests.

The last analysis was done on the complete problem instance. Since the class-based strategy resulted in the best solution for the small problem instance, we used this strategy again for the complete problem instance. The end result of this approach was that 245.58 hours were needed for the 8,512 retrieval requests, and 246.21 hours for the storage requests. This means that 491.79 hours were needed to handle all the requests.

5.2 Utilization analysis

In this section, we elaborate on the utilization rate of the system, and try to find an optimal value. In AS/R systems that have aisles with single-deep storage racks, all the stored products are directly accessible. The one product requires less time to reach than another, but in the end all the products are directly accessible. In a multi-deep AS/R system, like the one of Company X, this is not the case. Therefore, the amount of pallets and where these are stored significantly impacts the efficiency of the AS/R system, and hence we try to answer the following question in this section.

"What system utilization level yields the highest efficiency?"

Until now, we have assumed that it is most efficient to put a single type of product in storage lanes 1 and 4, and at most 2 different product types in storage lanes 2 and 3. Hereby, all the products remain accessible at all times. However, the next issue arises in the sense of the storage dates or the products. Ideally, ingredients that are stored the earliest must be retrieved first, before ingredients that are placed later. However, it sometimes happens that a product with a specific lot number must be retrieved, even when this is not the ingredient with the longest duration-of-stay (DOS). In this latter case we refer to the FEFO principle.

The first strategy we implemented was to store products based on their product name. By using this strategy, we ensure that each unique product is directly accessible. The first step was to pick the representative product list from Chapter 2, and collected the corresponding replenishment lot size for each product. Subsequently, we divided the maximum stock level of each product over the past 365 days by the replenishment lot size. The outcome is rounded up and we used that number as the number of orders for the specific product on the total order list. Thereafter, the total order list is randomized, to ensure that we evaluate different products with respect to allergens, heights, and replenishment size.

We tested 20 simulation runs to see what happens with the amount of products that can be stored. Since randomness is involved, we can set up a confidence interval to be, for example 95 percent, sure about the true value of number of products that can be stored. With these 20 simulation runs, we have a mean equal to 7,336 pallets, in combination with a standard deviation of 52. This means that we can say that we are 95 percent sure that the true value of the number of pallets that can be stored in the current composition lies between 7,317 and 7,355 pallets. This is equivalent to a maximum utilization level that lies between 84.30% and 84.74%.

The second strategy that we implemented was giving products a location based on lot number. This means that only products with the same lot number can be stored in the same storage lane, while before also different lot number could be stored in the same lane. In storage lanes 2 and 3, at most 2 different lot numbers can be stored. An advantage of storing product per lot number is that every product and lot number is accessible. Sometimes, products with specific lot numbers are requested, and this method ensures that all these products are directly accessible.

We started with an empty system again and tried to add a random lot size to the system one by one. The same approach was used as before, but now we created an unique ID for each replenishment within each unique product. This means that if 100 pallets of a certain product needs to be stored with a replenishment size of 30, we have 4 unique lot numbers for this product. 3 lot numbers have size 30, and 1 lot number has size 10. The lot sizes that occurred most in the list were of size 24, 26, 10, and 1. Eventually we calculated the average maximum utilization over the 20 runs, and this resulted in 7,166 pallets. With a standard deviation of 69 pallets, we are able to set up a 95% confidence interval, that ranges from 7,141 pallets as lower bound, and 7,191 pallets as upper bound. This is equivalent to a maximum utilization level that lies between 82.26% and 82.85%.

An additional analysis can be done on the utilization of the system, which incorporates swaps in order to create more storage space when no suitable location can be found anymore. This swap strategy is initiated when no storage location can be found, with as goal to create more storage space. This eventually results in a higher utilization level of the AS/R system. However, a disadvantage of this method is that it needs to be done on beforehand, since sometimes a lot of swaps are needed to create more storage space. With an already highly occupied system, this method is very hard to implement. With this method, we are able to store up to at most 7,845 pallets, which equals 90.38% of the maximum capacity.

Furthermore, we want to see what happens when we compare the time of relocating pallets with the time the shuttles could travel if they do not handle relocations. The main issue with relocating pallets is that it needs to find a suitable location, which could be difficult. Especially when the system is highly utilized, finding a free spot is more challenging than when the system is low utilized. If a free spot can be found, this spot must also be checked on allergen constraints, which means that it makes it even harder to find a suitable relocation. In Figure 9 below, we see 2 red cells located at location (30, 28, 1) and at location (26, 25, 1). When a pallet needs to be relocated from one of the locations to the other, it takes just as much time than compared to when the shuttle continues to drive forward. Thus, the locations that are marked green can be reached in the same time as the time it takes to relocate a pallet from one indicated location to the other.

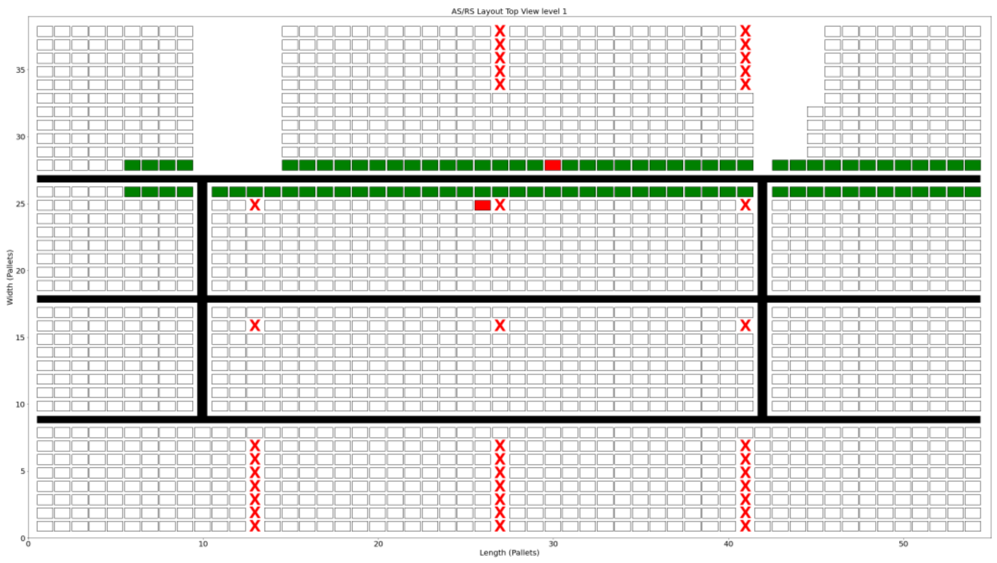

Figure 9: Relocations (red) compared to reachable locations (green)

From these results we can conclude that it would be efficient to store at most a single lot number in storage lanes 1 and 4, and at most 2 lot number in storage lanes 2 and 3. One of the main reasons for this is that it occasionally happens that specific lot numbers must be retrieved. Besides, Company X is planning to highly utilize the system with ingredients, and therefore it becomes much harder, if not impossible, to find a suitable location to relocate pallets. If no suitable location for relocating can be found, the pallet must be removed from the system in order to access the wanted products. This results in additional handling actions for the warehouse operators, which the stakeholders of Company X preferably want to prevent.

5.3 Analysis of results

In this section, we analyze all the results that are related to the AS/R system. In this way, we provide a clear overview on which methods and results are good, and which ones are less good. The following question is answered in this section:

"What are the results found and what do they say about the performance of the AS/RS?"

The first result that we obtained is related to the mathematical model formulated in Section 4.2.3. The total time that was needed to handle 70 storage requests and 40 retrieval requests equals 1.99 hours. That is equal to an average handling time per pallet of 1 minute and 5 seconds. This value is unrealistic low over the long run, and we can pinpoint a major reason for this. Because we are dealing with ingredients in the AS/R system, we have to take into account the expiry date. Products with a faster expiry date must be retrieved before products with a later expiry date. This means that eventually the products that are stored near the inbound station must be retrieved and transported to the outbound. Approximately twice as long is needed to retrieve a product near the inbound compared to retrieving it near the outbound. In reality, this is also dependent on whether a single or dual command cycle is used, but in the mathematical optimization model we only consider single cycles.

In Table 9 and Table 10 below, we only show results for the small problem instance. Later, the complete problem instance is handled. Both tables include the exact strategy, which is obtained by the SLAP optimization model of Section 4.2.3. The objective values for the other strategies are obtained by the simulation model. Table 9 shows the total travel time for the small problem instance. This time includes storage operations and retrieval operations. Table 10 on the other hand, shows only the retrieval time. The first column shows the strategy that is used, followed by the total time in Table 9, and the retrieval time in Table 10. Both of these values are transformed into averages.

The last two columns include the SA and VNS metaheuristic approaches that were used to improve the solution found so far. We know from literature that the quality of the meta heuristic solution is dependent on the quality of the initial solution. For the small problem instance, the class-based solution performed best for the retrieval times, and therefore we selected this storage strategy as starting point. The random storage strategy does not need a lot of computational effort for the small problem instance, and therefore we tested this storage strategy with both metaheuristic approaches as well. All values in the tables are in minutes.

Storage strategy	Total time	Average travel time	Average travel time after SA	Average travel time after VNS
Exact	119.33	1.08	-	-
Random	295.65	2.68	1.60	1.70
Class-based	233.25	2.12	1.30	1.45
Dedicated	210.53	1.91	-	-

Table 9: Total travel time results for the small problem instance (in minutes).

Storage strategy	Retrieval time	Average retrieval time	Average retrieval time after SA	Average retrieval time after VNS
Exact	46.75	1.17	-	-
Random	78.23	1.96	1.65	1.68
Class-based	60.53	1.51	1.25	1.35
Dedicated	65.35	1.63	-	-

Table 10: Retrieval time results for the small problem instance (in minutes).

The second set of results are related to the complete problem instance. This complete problem instance has 4,242 pallets stored initially, with 8,512 retrieval requests and 10,122 storage requests. The same format is used here as for Table 9 and Table 10. The major difference is that the complete problem instance handles 18,634 pallets, compared to 110 pallets for the small problem instance. This made us decide to change the total time and retrieval time format to hours. The average times are shown in minutes. The complete problem instance needed an extraordinary amount of time for the metaheuristic approach. For the class-based storage strategy, approximately 24 hours were needed to obtain the results with the SA approach, and approximately 18 hours were needed to obtain the results for the VNS method. This made us decide to only use the storage strategy that performed best before optimization. The exact solution is not shown in Table 11 and Table 12, since this was not possible with the SLAP optimization model.

Strategy	Total time (hours)	Average travel time	Average travel time after SA	Average travel time after VNS
Random	582.27	1.87	-	-
Class-based	504.46	1.62	1.58	1.58
Dedicated	534.63	1.72	-	-

Table 11: Total travel time results for the complete problem instance (in minutes, unless stated otherwise).

Strategy	Retrieval time (hours)	Average retrieval time	Average time after SA	Average time after VNS
Random	294.54	2.08	-	-
Class-based	249.24	1.75	1.72	1.73
Dedicated	257.06	1.80	-	-

Table 12: Retrieval time results for the complete problem instance (in minutes, unless stated otherwise).

A major difference between the small problem instance and complete problem instance can be noted and that is that the percentage improvements for the total travel times are significantly higher than for the retrieval times. On the other hand, the initial solution for the total travel time was also much higher than compared to the retrieval times, which leaves less space for improvement for the retrieval results. Besides, the metaheuristic approaches improve the small problem instance more than compared to the complete problem instance. One of the major reasons for this is that the influence of the expiry dates of the ingredients is higher for the complete problem instance than for the small problem instance. From the results we can see that the class-based storage strategy in combination with the SA method results in the best solution here, where the VNS method is good as well, but slightly worse than the SA method.

To conclude, 2 metaheuristic approaches were used to improve the initial solution. For the small problem instance, using the class-based storage strategy in combination with the SA method results in the most near optimal solution. The optimality gap for this result equals 7.14%. For the complete problem instance, the best strategy to use is the class-based storage strategy in combination with the SA approach as well.

5.4 Conclusion

Eventually when all 3 storage strategies were tested, we were able to implement an improvement phase to make the best storage strategy even better. Two metaheuristic approaches are used for this, of which the first one is Simulated Annealing, in short SA. The SA method uses a combination between exploration and exploitation, with as goal to escape from local optima. The second metaheuristic approach that was used was the Variable Neighborhood Search method, in short VNS. VNS uses diversification to change from neighborhood structure, in combination with intensification to find the local optimum of the current neighborhood structure. Both methods were tested to improve the small and complete problem instance. The SA method appears to be better for the smaller problem instance, where we were able to improve the solution up to an average retrieval time of 1.25 minutes. This is 7.14% above optimal, but the most likely reason for this is that in the simulation model we use, it is allowed to store only a single type of ingredient in storage lanes 1 and 4, while the optimization model does not take this into account. For the complete problem instance, both SA and VNS perform almost equally well. The improvements made by the metaheuristic approaches are better for the smaller problem instance compared to the complete problem instance.

6 Conclusion and recommendations

In this last chapter we provide a conclusion and recommendations for the stakeholders at Company X. In Section 6.1, a conclusion is provided in which an answer is given on the main research question. In Section 6.2, we provide specific recommendations to the stakeholders, after which we also discuss the limitations of this research in Section 6.3. Eventually, points for further research are mentioned as well in Section 6.4.

6.1 Conclusion

In this section, we give an answer to the proposed research question from Chapter 1. The main research question is:

"How do we design an optimal storage assignment for the automated storage/retrieval system of Company X such that the travel time efficiency is maximized?"

Company X is building a new warehouse in which it wants to use an AS/R system to automate processes. The efficiency of this system is dependent on the way it is used, and one of the decisions that must be made is the way in which products are stored in the system. To solve this problem, several methods and theoretical frameworks are proposed in literature. One of these methods is the analytic technique, that is able to solve optimization problems exactly. A major disadvantage of this method can be found in the computational effort that is needed for a successful implementation, and in addition this technique is only useful for small to medium-sized problem instances. On the other hand, simulation models and heuristics are an effective method to deal with problems that are of larger size.

Before the performance of the automated storage/retrieval system can be determined, we first need to analyze an important factor that influences this performance significantly. The production data plays an important role in what decisions must be made or not, to keep the operational efficiency of the AS/R system high. Specifically, the raw materials will occupy the system and therefore this becomes the focus point of the data collection phase. All the movements of ingredients during the past 365 days were collected, and the unwanted movements are omitted. The stakeholders at Company X were asked about their vision of this data to keep in line with their expectations. Eventually, we came to the conclusion that we only need the production movements of the ingredients. In other words, all the ingredients that are moved from a storage location to the batch area must be taken into account, and movements from one storage location to another not. This method ensures that we accurately visualize the number of movements for production per ingredient. Thereafter, we determined the maximum stock level over the past 365 days of each ingredient, with as goal to create a representative list with ingredients and stock levels. Since the expectation of Company X is to grow in the upcoming years, this method provides a good starting point for the expected stock levels in a few years.

When the relevant ingredients were selected, a representative replenishment list with storage requests and a list with retrieval requests is created in order to be able to determine the performance of the AS/R system. An employee from the planning department was asked about the production schedule, and provided insights in the production cycles. In a period of 13 weeks, all the products produced by Company X are produced at least once. This means that this period is a representative period in which all storage and retrieval requests can be gathered. At this stage, we were able to set up the optimization model, but the problem instance was too complex to evaluate exactly.

Subsequently, we created a simulation model that is able to determine the travel times of the shuttles in the AS/R system precisely. After trying to store all the 7,764 pallets from the representative product list, this was not feasible as well. The main reason for this was that the current product composition with allergens and heights was too restrictive. Despite that from this situation a conclusion can be drawn on the limitations of the AS/R system, it is not the desired outcome. The initial product list with stock levels of each ingredient was reduced from 7,764 to 4,242 pallets. This resulted in an feasible starting point. The next steps of the simulation model are the initialization of the storage and retrieval requests, processing of the requests, and eventually the calculation of the objective function. The 4,242 pallets represent the stock level at the first day of the production cycle of 13 weeks. We implemented 3 storage strategies that were frequently used in literature. These strategies are random, class-based, and dedicated. For the complete problem instance, the results are shown in Table 13 below.

Strategy	Average total time	Average retrieval time	Average storage time
Random	1.87	2.08	1.70
Class-based	1.62	1.75	1.52
Dedicated	1.72	1.80	1.65

Table 13: Average results for the complete problem instance (in minutes).

From the results above, we can conclude that the class-based storage strategy performs best, by performing 6.19% better than the dedicated storage strategy, and 15.46% better than the random storage strategy. To improve the results, 2 metaheuristic methods were used. The Simulated Annealing approach is the first one, and this method manages to bring down the average total time from 1.62 minutes to 1.58 minutes. For the average retrieval time, this is decreased from 1.75 minutes to 1.72 minutes. The second approach that was used is the Variable Neighborhood Search approach. After using this method, the average total time is decreased from 1.62 minutes to 1.58 minutes, and the average retrieval time from 1.75 minutes to 1.73 minutes.

The percentage decrease for the 2 metaheuristic approaches is higher for the smaller problem instance than for the complete problem instance. A likely reason for this is that the system has a lower utilization rate and thereby having more possibilities for improvement. A second reason is that the ingredients must be retrieved based on lot number, meaning that ingredients with a shorter expiry date are preferred over ingredients with a longer expiry date. Having an initial stock level for the complete problem instance of 4,242 pallets, in combination with 8,512 retrieval requests and 10,122 storage requests, at least 4,270 pallets must be retrieved that are on the storage list as well. Therefore, not only the initial state influences the retrieval times, but also the way in which products are stored. For the small problem instance, the retrieval times are only influenced by the initial state and not by any storage requests.

Last, an analysis was done on the occupation rate of the AS/R system, since we initially were not able to store 7,764 pallets. The storage preferences of having at most 1 type of ingredient in storage lanes 1 and 4 and at most 2 different ingredients in storage lanes 2 and 3 are taken into account. The first analysis was done on storing ingredients according to their ingredient ID, and after 20 simulation runs and the current product composition, we can say with 95% certainty that the maximum achievable utilization rate lies between 84.30% and 84.74% if the pallets are stored based on ingredient ID. We did the same analysis but then storing on lot number, which resulted in a 95 percent confidence interval of utilization rate ranging from 82.26% to 82.85%.

6.2 Recommendations

This section gives a summation of recommendations, based on the findings and conclusions from this research

- For a highly utilized system the class-based storage strategy performs the best, in combination with using 3 classes.
- With a highly utilized system, it is not necessary to use the metaheuristic approach to improve the travel time.
- With a low utilized system, the dedicated storage strategy works best, where the closest storage lanes are selected first for storage.
- Use at most a single lot number in storage lanes 1 and 4, and at most 2 lot numbers in storage lanes 2 and 3.
- With a highly utilized system, it is not necessary to update the product classes on a regular basis. Updating the classes might be useful when a production cycle is finished, thus once every quarter.
- Investigate the validity of the collected data. During the data collection process, several thoughts about the quality of the data were shared, and all stakeholders should agree on the used data.
- Reconsider the allergen storage constraints, because by relaxing this constraint an even higher system utilization can be reached.

6.3 Limitations

It is not unimportant to mention some limitations that are linked to this research, in order to show awareness on the decisions made and which affects these decisions had.

- First of all, the improvement phases where the SA method or the VNS method are used takes significant time to evaluate. In practice, this is not really convenient since sometimes quick solutions are needed for unforeseen circumstances.
- Second, during the research only static data was used. This means that the height and stock levels do not change, which could be the case in reality.
- The AS/R system of Company X is still under construction, which limits us to test certain results or conclusions in reality.
- This research does not take into account the way the software of the AS/R communicates with the software of Company X. In order to implement conclusions or recommendations from this research, it might be needed to connect the simulation model in python to the software of the AS/R system.

6.4 Further Research

In the following paragraphs we look ahead on what can be done more to improve the quality of the results, and increase the efficiency of the AS/R system even further if possible. First of all, during the research different stakeholders were asked and involved in the data collection process, with as goal to create a complete and accurate view of reality. We assumed that the collected data is correct, and therefore the validation of collected data is not in the scope of this research. However, in order to increase the accuracy of the results, this data must be validated in such a way that all the stakeholders agree with each other.

Second, we collected data in such a way that it represents the reality as accurate as possible. This data includes movement numbers, incoming and outgoing pallet numbers, lot sizes, pallet heights, and stock levels. All the data that was used is not updated in the meantime, and to increase the accuracy, this can be updated periodically. When this data list is frequently updated and very accurate, a correlation analysis can be done as well. In this research, we assumed that only a single product can be placed in storage lane 1 and 4, and at most 2 in storage lanes 2 and 3. However, some types of ingredients might be on the same order list all the time, and can therefore, in the appropriate quantities, be placed in the same storage lane.

On the other hand, when different types of products are stored in a single storage lane that is only accessible from a single side, and an urgent requests comes in for ingredients at the end of the storage lane, the products in front must be relocated. It is experienced that relocating ingredients take a lot of time, especially due to the wheel set change time that is needed to make a turn. Furthermore, finding a suitable location that does not violate the storage rules can be difficult and time consuming, dependent on the distance between the original location and the location after relocation. Especially when the system is highly utilized, it can become more and more complex to find a suitable location for the relocating process.

Besides, due to restrictions in available data, we used a single pallet height for each product. The height that was used corresponds to the pallet height at the moment when the pallet arrives at the warehouse for the first time. It is possible that during the picking process only not the entire pallet is needed, but only a part of it. As a result, the height of the pallet is becoming lower than it was when it came in for the first time, and can therefore potentially be placed back in the system at a lower level than were it was located initially. Placing pallets at a lower level can increase the efficiency, since it is more likely to have a lower storage or retrieval time, especially for the lowest level.

Last, as also mentioned by Yang et al. (2013), the efficiency of an AS/R system is maximized when SLAP is combined with the storage and retrieval scheduling problem. This is called the location assignment and storage/retrieval scheduling problem, in short LASRSP. What happens when the storage and retrieval request schedule is optimized, certain requests are handled earlier or later, dependent on the outcomes. Since Company X is producing according to a strict planning, we cannot simply change the order of requests. This was also one of the main reasons why this is not included in the scope of this research. However, a further research can assess the feasibility of changing the request schedule in such a way that even a higher efficiency can be achieved.

References

Aarts, E., Korst, J., & Michiels, W. (2005). Simulated Annealing. In E. K. Burke, & G. Kendall, *Search Methodologies Introductory Tutorials in Optimization and Decision Support Techniques* (pp. 187-211). Springer.

Confidential

Ahmed, H., & Glasgow, J. (2012). *Swarm Intelligence: Concepts, Models and Applications.* School Of Computing, Queens University Technical Report.

Arifin, S. N., & Madey, G. R. (2019). Verification, Validation, and Replication Methods for Agent-Based Modeling and Simulation: Lessons Learned the Hard Way! In L. Yilmaz, *Concepts and Methodologies for Modeling and Simulation* (pp. 217-243). Auburn, AL: Springer.

Bahrami, B., Piri, H., & Aghezzaf, E.-H. (2019). Class-based storage location assignment - an overview of the literature. *16th International Conference on Informatics in Control, Automation and Robotics*, pp. 390-397.

Bessenouci, H. N., Sari, Z., & Ghomri, L. (2012). Metaheuristic based control of a flow rack automated storage retrieval system. *Journal of Intelligent Manufacturing*, 1-10.

Blum, C., & Roli, A. (2001). Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. *ACM Computing Surveys*, 268-308.

Brezovnik, S., Gotlih, J., Balič, J., Gotlih, K., & Brezočnik, M. (2015, February 24). Optimization of an Automated Storage and Retrieval Systems by Swarm Intelligence. *Procedia Engineering*, pp. 1309-1318.

Caporossi, G., Hansen, P., & Mladenovic, N. (2016). Variable Neighborhood Search. In P. Siarry, *Metaheuristics* (pp. 77-98). Paris: Springer.

Caron, F., & Marchet, G. (2010). Routing policies and COI-based storage policies in picker-to-part systems. *International Journal of Production Research*, 713-732.

Csanády, E., Kovács, Z., Magoss, E., & Ratnasingam, J. (2019). Principles of Optimization. In E. Csanády, Z. Kovács, E. Magoss, & J. Ratnasingam, *Optimum Design and Manufacture of Wood Products* (pp. 147-213). Cham: Springer Nature Switzerland AG.

Curry, R. (2013). *A Duration-of-Stay Storage Policy in the Physical Internet.* Fayetteville, Arkansas: University of Arkansas.

De Koster, R. (2008). Warehouse Assessment in a Single Tour. In R. Manzini, *Warehousing in the Global Supply Chain* (pp. 457-473). London: Springer.

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

Dorigo, M., & Stützle, T. (2004). *Ant Colony Optimization.* Cambridge, Massachusetts: MIT Press.

Eynan, A., & Rosenblatt, M. J. (1993). An interleaving policy in automated storage/retrieval systems. *International Journal of Production Research*(1), pp. 1-18.

Fandi, W., Kouloughli, S., & Ghomri, L. (2022). Multi-shuttle AS/RS dimensions optimization using a genetic algorithm—case of the multi-aisle configuration. *The International Journal of Advanced Manufacturing Technology*, 1219-1236.

Floudas, C. A., & Lin, X. (2005, October). Mixed Integer Linear Programming in Process Scheduling: Modeling, Algorithms, and Applications. *Annals of Operations Research*, pp. 131-162.

Fumi, A., Scarabotti, L., & Schiraldi, M. M. (2013, June 12). Minimizing Warehouse Space with a Dedicated Storage Policy. *International Journal of Engineering Business Management, 5*.

Fumi, A., Scarabotti, L., & Schiraldi, M. M. (2013, June 12). Minimizing Warehouse Space with a Dedicated Storage Policy. *International Journal of Engineering Business Management, 5*.

Goetschalckx, M. (2012). Storage Systems and Policies. In R. Manzini, *Warehousing in the Global Supply Chain* (pp. 31-51). London: Springer.

Goetschalckx, M., & Ratliff, H. D. (1990). Shared Storage Policies Based on the Duration Stay of Unit Loads. *management Science*, 1120-1132.

Gogna, A., & Tayal, A. (2013). Metaheuristics: review and application. *Journal of Experimental & Theoretical Artificial Intelligence, 25*(4), 503-526.

Gogna, A., & Tayal, A. (2013). Metaheuristics: review and application. *Journal of Experimental & Theoretical Artificial Intelligence*, 503-526.

Good manufacturing practice. (2024, March 20). Retrieved from www.ema.europa.eu: https://www.ema.europa.eu/en/human-regulatory-overview/research-development/compliance-research-development/good-manufacturing-practice#:~:text=Good%20manufacturing%20practice%20(GMP)%20describes,European%20Union%20(EU)%20level.

Graves, S. C., Hausman, W. H., & Schwarz, L. B. (1977, May). Storage-Retrieval Interleaving in Automatic Warehousing Systems. *Management Science*, pp. 935-945.

Guenov, M., & Raeside, R. (1992). Zone shapes in class based storage and multicommand order picking when storage/retrieval machines are used. *European Journal of Operational Research*, pp. 37-47.

Ha, Y., & Chae, J. (2019). A decision model to determine the number of shuttles in a tier-to-tier SBS/RS. *International Journal of Production Research*, 963-984.

Halpin, T., & Morgan, T. (2008). Relational Languages. *The Morgan Kaufmann Series in Data Management Systems*, 527-635.

Hansen, P., & Mladenović, N. (2001). Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, 449-467.

Heragu, S. S., Du, L., Mantel, R. J., & Schuur, P. C. (2005, January 15). Mathematical model for warehouse design and product allocation. *International Journal of Production Research*, pp. 327-338.

Hole, G., Hole, A. S., & McFalone-Shaw, I. (2021). Digitalization in pharmaceutical industry: What to focus on under the digital implementation process? *International Journal of Pharmaceutics: X*.

Khan, M. N., Sinha, A. K., & Anand, A. (2023). Use of metaheuristics in industrial development and their future perspectives. *Comprehensive Metaheuristics*, 195-202.

Kouloughli, S., & Sari, Z. (2015). Multi-aisle AS/RS dimensions optimization for cycle time minimization. *The International Journal of Advanced Manufacturing Technology*, 675-692.

Kusrini, E., Novendri, F., & Helia, V. (2018). Determining key performance indicators for warehouse performance measurement - a case study in construction materials warehouse. *The 2nd International Conference on Engineering and Technology for Sustainable Development*.

Law, A. M. (2015). *Simulation Modeling and Analysis.* New York, NY: McGraw-Hill.

Lee, H. F. (1997). Performance analysis for automated storage and retrieval systems. *IIE Transactions*, 15-28.

Lee, M. -K., & Elsayed, E. A. (2005). Optimization of warehouse storage capacity under a dedicated storage policy. *International Journal of Production Research*, pp. 1785-1805.

Leeftink, G. (2022). *Lecture 11 - Improvement heuristics II.* Retrieved from www.canvas.utwente.nl: https://canvas.utwente.nl/courses/10786/pages/lecture-11-improvement-heuristics-ii?module_item_id=343809

Leng, J., Sha, W., Wang, B., Zheng, P., Zhuang, C., Liu, Q., . . . Wang, L. (2022, October). Industry 5.0: Prospect and retrospect. *Journal of Manufacturing Systems*, pp. 279-295.

Liviu, I., Ana-Maria, T., & Emil, C. (2009). Warehouse Performance Measurement - A Case Study. *Annals of Faculty of Economics*, 307-312.

Man, X., Zheng, F., Chu, F., Liu, M., & Xu, Y. (2019, April 10). Bi-objective optimization for a two-depot automated storage/retrieval system. *Annals of Operations Research*, pp. 243-262.

Manzini, R. (2011). *Warehousing in the Global Supply Chain.* London: Springer London.

Meller, R. D., & Mungwattana, A. (2005). AS/RS dwell-point strategy selection at high system utilization: A simulation study to investigate the magnitude of the benefit. *International Journal of Production Research*, 5217-5227.

Mirjalili, S. (2018). *Evolutionary Algorithms and Neural Networks.* Cham: Springer International Publishing.

Our Brands. (2023). Retrieved from www.CompanyXnutrition.com: https://www.CompanyXnutrition.com/our-brands

Park, B. C. (1987). Closest Open Location Rule in AS/RS. *Journal of the Korean Institute of Industrial Engineers, 13*.

Park, B. C. (2011). Order Picking: Issues, Systems and Models. In R. Manzini, *Warehousing in the Global Supply Chain* (pp. 1-30). London: Springer.

Pisinger, D., & Ropke, S. (2010). Large Neighborhood Search. In M. Gendreau, & J.-Y. Potvin, *Handbook of Metaheuristics* (pp. 399-420). Springer.

Randhawa, S. U., & Shroff, R. (1995). Simulation-based design evaluation of unit load automated storage/retrieval systems. *Computers & Industrial Engineering*, 71-79.

Roodbergen, K. J. (2012). Storage Assignment for Order Picking in Multiple-Block Warehouses. In R. Manzini, *Warehousing in the Global Supply Chain* (pp. 139-158). London: Springer-Verlag London Limited.

Roodbergen, K., & Vis, I. (2009, April 16). A survey of literature on automated storage and retrieval systems. *European Journal of Operational Research*, pp. 343-362.

Savory, P. A., & Mackulak, G. T. (1994). The Science of Simulation Modeling. *International Conference on Simulation in Engineering Education*, 115-119.

Schuur, P. C. (2015). The worst-case performance of the Cube per Order Index slotting strategy is infinitely bad - A technical note. *International Journal of Production Economics*, 801-804.

Silva, A., Roodbergen, K. J., Coelho, L. C., & Darvish, M. (2022, October). Estimating optimal ABC zone sizes in manual warehouses. *International Journal of Production Economics, 252*.

Sivanandam, S. N., & Deepa, S. N. (2008). *Introduction to Genetic Algorithms.* Springer.

Staudt, F. H., Alpan, G., Mascolo, M. D., & Toboada Rodriguez, C. M. (2015). Warehouse performance measurement: a literature review. *International Journal of Production Research*, 5524-5544.

Uranucci, L. (2018, September). Limits and potentials of Mixed Integer Linear Programming methods for optimization of polygeneration energy systems. *Energy Procedia*, pp. 1199-1205.

van den Berg, J. P., & Gademann, N. (2000, April). Simulation study of an automated storage/retrieval system. *International Journal of Production Research* , pp. 1339-1356.

Vasili, M., Tang, S. H., & Vasili, M. (2014). Automated Storage and Retrieval Systems: A Review on Travel TIme Models and Control Policies. *Warehousing in the Global Supply Chain*, 159-209.

Xu, X., Zhao, X., Zou, B., & Li, M. (2018). Optimal dimensions for multi-deep storage systems under class-based storage policies. *Cluster Computing* , 861-875.

Yang, P., Miao, L., Xue, Z., & Qin, L. (2013, November 17). An integrated optimization of location assignment and storage/retrieval scheduling in multi-shuttle automated storage/retrieval systems. *Journal of Intelligent Manufacturing*, pp. 1145-1159.

Yang, P., Miao, L., Xue, Z., & Ye, B. (2015, July). Variable neighborhood search heuristic for storage location assignment and storage/retrieval scheduling under shared storage in multi-shuttle automated storage/retrieval systems. *Transportation Research Part E: Logistics and Transportation Review*, pp. 164-177.

Yang, P., Yang, K., Qi, M., Miao, L., & Ye, B. (2017). Designing the optimal multi-deep AS/RS storage rack under full turnover-based storage policy based on non-approximate speed model of S/R machine. *Transportation Research Part E: Logistics and Transportation Review*, 113-130.

Yu, Y., & de Koster, R. B. (2009). Optimal zone boundaries for two-class-based compact three-dimensional automated storage and retrieval systems. *IIE Transactions*, 194-208.

Zhou, G., & Mao, L. (2010). Design and Simulation of Storage Location Optimization Module in AS/RS Based on FLEXSIM. *International Journal of Intelligent Systems and Applications*, 33-40.

Appendix A – Products of Company X

Confidential

Appendix B – Multi-deep AS/RS layout of Company X

Confidential

Appendix C – Analysis of AS/RS layout

Confidential

Appendix D – Complete optimization model

In this Appendix D, we show the complete optimization model that most accurately represents the real system. However, this still remains a simplified version, since in reality stochasticity's and unexpected situations could occur. The main difference is that in this model, the number of operation cycles, indicated by K, is much larger than in the simplified version. By using the operation cycle indicator k, we are able to prefer certain operations over others, and thereby making the model a more accurate representation of the reality.

Sets

S	Shuttles $s \in S$
I	Storage positions in length $i \in I$
J	Storage positions in width $j \in J$
Z	Storage level $z \in Z$
A	product allergen $a \in A$
P	Set of products $p \in P$
RP_k	Products that must be retrieved $RP \subseteq P$ in cycle $k \in K$
SP_k	Products that must be stored $SP \subseteq P$ in cycle $k \in K$
K	Set of all operation cycles in the planning horizon
L	Set of storage lanes $l \in L$
N	Shuttle level combinations $n \in N$
J_l	Set of storage positions $j \in J$ in lane $l \in L$
F	Type $f \in F$ of a storage lane
RJ_l^f	Range of lane $l \in L$ of type $f \in F$
U	Unavailable storage locations $u \in U$

Parameters

h_p	height in millimeters of product $p \in P$
p_a	allergen $a \in A$ of product $p \in P$
h_z	maximum storage height of level z in millimeters $z \in Z$
c_z	total pallet capacity for level $z \in Z$
$st_{i,j,z}$	time to travel from inbound to location i, j at level z, and back to the inbound
$rt_{i,j,z}$	time to travel from outbound to location i, j at level z, and back to the outbound
TT	total time for all requests in seconds
M	A large enough number

H_p^k Number of products $p \in P$ stored during cycle $k \in K$

Variables

$q_{i,j,z,p}^k \quad = \begin{cases} 1 \text{ if location } i, j \text{ at level } z \text{ is occupied with product } p \text{ during cycle } k \\ 0 \text{ otherwise} \end{cases}$

$x_{i,j,z,p}^k \quad = \begin{cases} 1 \text{ if product } p \text{ is retrieved from location } i, j \text{ at level } z \text{ during cycle } k \\ 0 \text{ otherwise} \end{cases}$

$o_{i,l,z,p}^k \quad = \begin{cases} 1 \text{ if lane } l \text{ at location } i \text{ at level } z \text{ is occupied with product } p \text{ during cycle } k \\ 0 \text{ otherwise} \end{cases}$

$d_{i,j,z,p}^k \quad = \begin{cases} 1 \text{ if product } p \text{ is stored at location } i, j \text{ at level } z \text{ during cycle } k \\ 0 \text{ otherwise} \end{cases}$

$y_{i,j,z,s} \quad = \begin{cases} 1 \text{ if location } i, j \text{ at level } z \text{ can reached by shuttle } s \\ 0 \text{ otherwise} \end{cases}$

$b_{i,j,z}^k \quad = \begin{cases} 1 \text{ if location } i, j \text{ at level } z \text{ is accessible from at least one side during cycle } k \\ 0 \text{ otherwise} \end{cases}$

Objective Function

$$Min\, TT = \sum_{k=1}^{K} \left(\sum_{p=1}^{RP_k} \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{z=1}^{Z} rt_{i,j,z} * x_{i,j,z,p}^k + \sum_{p=1}^{SP_k} \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{z=1}^{Z} st_{i,j,z} * d_{i,j,z,p}^k \right)$$

Constraints

$q_{i,j,z,p}^k * h_p \leq h_z$ $\quad \forall i, \forall j, \forall z, \forall p, \forall k$ (1)

$q_{i,j,z,p_a}^k + q_{i,j,z',p'_a}^k \leq 2$ $\quad \forall i, \forall j, \forall k, \forall z' > z, a \in \{2, 3, 4\}, a' \in \{1\}$ (2)

$q_{i,j,z,p_a}^k + q_{i,j,z',p'_a}^k \leq 1$ $\quad \forall i, \forall j, \forall k, \forall z' > z, a \in \{1, 2, 3, 4\}, a' \in \{2, 3, 4\}, a \neq a'$ (3)

$y_{i,j,z,s} = 1$ $\quad \forall i, \forall j, \forall (z, s) \in N$ (4)

$q_{i,j,z,p}^k = 0$ $\quad (i, j, z) \in U, \forall p, \forall k$ (5)

$\sum_{p=1}^{P} q_{i,j,z,p}^k \leq 1$ $\quad \forall i, \forall j, \forall z, \forall k$ (6)

$\sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{z=1}^{Z} q_{i,j,z,p}^k = H_p^k$ $\quad \forall k, \forall p$ (7)

$x_{i,j,z,p}^k \leq q_{i,j,z,p}^k$ $\quad \forall i, \forall j, \forall z, \forall k, \forall p \in RP_k$ (8)

$M * \left(1 - d_{i,j,z,p'}^k\right) \geq \sum_{p=1}^{P} q_{i,j,z,p}^k$ $\quad \forall i, \forall j, \forall z, \forall p' \in SP_k, k \geq 1$ (9)

$\sum_{s=1}^{S} y_{i,j,z,s} \geq x_{i,j,z,p}^k$ $\quad \forall i, \forall j, \forall z, \forall k, \forall p \in RP_k$ (10)

$$\sum_{s=1}^{S} y_{i,j,z,s} \geq d_{i,j,z,p}^{k} \qquad \forall i, \forall j, \forall z, \forall k, \forall p \in SP_k \tag{11}$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{z=1}^{Z} x_{i,j,z,p}^{k} = 1 \qquad \forall k, \forall p \in RP_k \tag{12}$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{z=1}^{Z} d_{i,j,z,p}^{k} = 1 \qquad \forall k, \forall p \in SP_k \tag{13}$$

$$\sum_{p=1}^{P} o_{i,l,z,p}^{k} \leq 1 \qquad \forall i, \forall z, \forall k, \forall l \in \{1,4\} \tag{14}$$

$$\sum_{p=1}^{P} o_{i,l,z,p}^{k} \leq 2 \qquad \forall i, \forall z, \forall k, \forall l \in \{2,3\} \tag{15}$$

$$\sum_{j \in J_l} q_{i,j,z,p}^{k} \leq M * o_{i,l,z,p}^{k} \qquad \forall i, \forall z, \forall l, \forall p, \forall k \tag{16}$$

$$\sum_{j \in J_l} q_{i,j,z,p}^{k} \geq o_{i,l,z,p}^{k} \qquad \forall i, \forall z, \forall l, \forall p, \forall k \tag{17}$$

$$\sum_{p'=1}^{P}\sum_{j' \in RJ_l^1} q_{i,j',z,p}^{k} \leq M\left(1 - d_{i,j,z,p}^{k}\right) \qquad \forall i, \forall z, \forall p, \forall k, \forall l \in \{1,4\}\ \forall j \in J_l \tag{18}$$

$$\sum_{p'=1}^{P}\sum_{j' \in RJ_l^1} q_{i,j',z,p}^{k} \leq M\left(1 - x_{i,j,z,p}^{k}\right) \qquad \forall i, \forall z, \forall p, \forall k, \forall l \in \{1,4\}\ \forall j \in J_l \tag{19}$$

$$d_{i,j,z,p}^{k} \leq b_{i,j,z}^{k} \qquad \forall i, \forall j \in \{J_2, J_3\}, \forall z, \forall k, \forall p \in SP_k \tag{20}$$

$$x_{i,j,z,p}^{k} \leq b_{i,j,z}^{k} \qquad \forall i, \forall j \in \{J_2, J_3\}, \forall z, \forall k, \forall p \in RP_k \tag{21}$$

$$\sum_{p=1}^{P}\sum_{j' \in RJ_l^1} q_{i,j',z,p}^{k} \leq M * b_{i,j,z}^{k} \qquad \forall i, \forall z, \forall p, \forall k, \forall l \in \{2,3\}, \forall j \in J_l \tag{22}$$

$$\sum_{p=1}^{P}\sum_{j' \in RJ_l^2} q_{i,j',z,p}^{k} \leq M\left(1 - b_{i,j,z}^{k}\right) \qquad \forall i, \forall z, \forall p, \forall k, \forall l \in \{2,3\}, \forall j \in J_l \tag{23}$$

$$\sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{z=1}^{Z} q_{i,j,z,p}^{0} = P_p^0 \qquad \forall p \tag{24}$$

$$q_{i,j,z,p}^{k} = q_{i,j,z,p}^{k-1} + d_{i,j,z,p}^{k-1} - x_{i,j,z,p}^{k-1} \qquad \forall i, \forall j, \forall z, \forall p, \forall k \geq 1 \tag{25}$$

$$q_{i,j,z,p}^{k} \in \{0,1\} \qquad \forall i \in I, j \in J, z \in Z, p \in P, \forall k \in K \tag{26}$$

$$x_{i,j,z,p}^{k} \in \{0,1\} \qquad \forall i \in I, j \in J, z \in Z, p \in RP_k, \forall k \in K \tag{27}$$

$$o^{k}_{i,l,z,p} \in \{0,1\} \qquad \forall i \in I, l \in L, z \in Z, p \in P, \forall k \in K \qquad (28)$$

$$d^{k}_{i,j,z,p} \in \{0,1\} \qquad \forall i \in I, j \in J, z \in Z, p \in SP_k, \forall k \in K \qquad (29)$$

$$y_{i,j,z,s} \in \{0,1\} \qquad \forall i \in I, j \in J, z \in Z, s \in S \qquad (30)$$

Explanation of constraints

The first constraint ensures that if a pallet with product p is placed at location i, j at level z, the height of this pallet cannot exceed the maximum storage height of level z. Constraint 2 only allows the same allergen types above each other, when $p_a{}'$ is not equal to 1 (Non-allergen). Constraint 3 allows products classified as non-allergens to be placed above allergen-products. The exact rules are described in Section 2.2.1 Constraints. For constraint 4, the shuttles are assigned to a predetermined level. This is based on the advice given by the supplier, since the requested number of cycles per hour is not high enough for the shuttles to move between different levels. However, we have 4 shuttles for 5 levels, meaning that one shuttle must serve two levels. These shuttle and level combinations are given by set N. Moreover, not every location in the AS/R system can be used for pallet storage. The unavailable storage locations are given by set U, and constraint 5 guarantees that no products can be stored at these locations. Moreover, constraint 6 ensures that all locations cannot be occupied by more than one pallet at the same time. Constraint 7 keeps track of the amount of pallets stored of a specific product during a specific cycle k. Constraint 8 allows a product to be picked from a location only if that product is actually stored at that specific location.

Constraint 9 is making sure that the location is empty if a product is stored at that location. Constraint 10 and 11 ensure that locations can be reached by at least one shuttle when a product must be retrieved or stored respectively. Besides, constraint 12 and 13 ensure that all the retrieval requests and storage requests are fulfilled. Eventually, storage lanes of type 1 can only store a single type of product at the same time, while storage lanes of type 2 can store up to 2 different products at the same time. Constraints 14 and 15 indicate this respectively, but with the help of constraints 16 and 17. These two constraints help to indicate whether a certain product p is stored in a specific storage lane l.

Constraints 18 and 19 ensure that when a storage request or retrieval request is performed in storage lane 1 or 4, this location is accessible. When the first position seen from the main aisle is occupied and the positions behind it are empty, these empty positions are not directly accessible for storage operations. The same yields for retrieval requests, where the specific location must be accessible as well in order to be able to perform this retrieval request. Constraints 20, 21, 22, and 23 ensure that positions located in storage lanes 2 and 3 are accessible from at least one side. Constraint 24 indicates that a starting stock level must be stored. Eventually, constraint 25 is the inventory balance equation, that ensures that the inventory level of a certain product is updated when a storage or retrieval request is performed. Constraints 26, 27, 28, 29, and 30 are sign constraints, and can only take value 0 or 1.

Model input data

This section is an extension on the previous section, primarily to discuss the relevant data that is needed as input for the optimization model. Below, a summation is made with the used input data. To indicate the storage positions, we use set I for positions in length, and set J for positions in width. If we look at **Error! Reference source not found.**, the first value of set I starts at the left hand side and e nds with the last value of set I at the right hand side. The same applies for set J, where the first value of this set starts at the bottom and the last value ends at the top. If we apply this in practice, we see that coordinate (i, j) = (1,1) is located at the bottom left hand side, and coordinate (i, j) = (54,38) is located at the top right hand side. Additionally, we include level z as well, resulting in a format with 3 coordinates: (i, j, z). Besides, for the allergen set A, we have a non-allergen, a milk allergen, a soy allergen, and a cereal allergen as possible values. To simplify the use of this set, we use index 1 until 4 to indicate the allergen type.

Sets

Shuttles $S = \{1, 2, 3, 4\}$

Storage positions in length $I = \{1, 2, \dots, 54\}$

Storage positions in width $J = \{1, 2, \dots, 38\}$

Storage level $Z = \{1, 2, \dots, 5\}$

product allergen $A = \{1\ (Non), 2\ (Milk), 3\ (Soy), 4\ (Cereals)\}$

The next few sets are related to the ingredients that are stored and/or retrieved. The ingredients are also called products, and all have an unique name. However, for sake of simplicity we just use numbers. For the storage requests and retrieval requests, these are both sets with products from the original product set P, and are therefore both called multiset RP and SP. Again, for sake of simplicity we use numbers instead of product names. This means that we have 8,512 retrieval requests, and 10,122 storage requests in total.

Stored products $P = \{1, 2, \dots, 106\}$

Products that must be retrieved $RP = \{1, 2, \dots, 8{,}512\}$

Products that must be stored $SP = \{1, 2, \dots, 10{,}122\}$

To incorporate operation cycle k, we have to combine RP and SP into a single set. This is due to the fact that the shuttles can only carry a single pallet at a time, and therefore can only store or retrieve a single product at a time. A single operation cycle can therefore only include only one storage request or one retrieval request. If both RP and SP are added together, we end up with 18,634 operation cycles.

Operation cycles $K = \{1, 2, \dots, 18{,}634\}$

Set of storage lanes $L = \{1, 2, 3, 4\}$

Shuttle level locations $N = \{(1{,}1), (2{,}2), (3{,}3), (4{,}4), (5{,}4)\}$

Storage positions j in lane l. $J_1 = \{1, 2, 3, 4, 5, 6, 7, 8\}$, $J_2 = \{10, 11, 12, 13, 14, 15, 16, 17\}$, $J_3 = \{19, 20, 21, 22, 23, 24, 25, 26\}$, $J_4 = \{28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38\}$

For the storage lane, two types are present in the AS/R system of Company X. The first one is a storage lane that is accessible by only a single side, and the second one is a storage lane that is accessible by two sides. Storage lane 1 and 4 are only accessible by a single side, and are therefore classified as type 1 lane, and storage lane 2 and 3 are accessible by two sides, and are therefore classified as type 2 lanes.

Type of storage lane $F = \{1, 2\}$

Range of lane $l \in L$ of type $f \in F, RJ_l^f$:

$RJ_1^1(j') = \{\, j'' \mid j' \leq j'' \leq 8 \,\}$

$RJ_4^1(j') = \{j'' \mid j' \leq j'' \leq 28\}$

$RJ_4^2(j') = RJ_1^1(j') = \{\}$

$RJ_2^1(j') = \{j'' \mid 10 \leq j'' \leq j'\}$

$RJ_2^2(j') = \{j'' \mid j' \leq j'' \leq 17\}$

$RJ_3^1(j') = \{j'' \mid 19 \leq j'' \leq j'\}$

$RJ_3^2(j') = \{j'' \mid j' \leq j'' \leq 26\}$

For the unavailable storage locations, denoted by U, a short explanation is desired. The size of set U equals 1,580, meaning that we have 1,580 unavailable storage locations in total. We can check whether this approach is correct, by multiplying the maximum number of I, J, and Z, and subtract 1,580 from that. The multiplication of these three numbers equal 10,260, and after subtracting 1,580 from that we end up with 8,680. This is exactly the maximum capacity of the AS/R system, and hence we have verified that the size of set U is correct. To present 1,580 values in an efficient way, we have created Table D1**Error! Reference source not found.**. In this table, 4 columns are presented with several rows c ontaining sets. In order to find all unavailable coordinates, we take the Cartesian Product of i_k, j_k, and z_k for all k. The Cartesian Product or Unrestricted Join of sets A' and B' is denoted by $A' \times B'$ and results in a set of all ordered pairs (a', b') such that a' belongs to A' and b' to B' (Halpin & Morgan, 2008). In our case, we use 3 sets at a time for the Cartesian Product, and create coordinates with format (i, j, z).

Number n	**Length i**	**Width j**	**Level z**
1	$\{1, 2, \dots, 54\}$	$\{9, 18, 27\}$	$\{1, 2, 3, 4, 5\}$
2	$\{10, 42\}$	$\{10, 11, \dots, 26\} \backslash \{18\}$	$\{1, 2, 3, 4, 5\}$
3	$\{13, 27, 41\}$	$\{1, 2, \dots, 7\}, \{16, 25\}$	$\{1, 2, 3, 4, 5\}$
4	$\{27, 41\}$	$\{34, 35, \dots, 38\}$	$\{1, 2, 3, 4, 5\}$
5	$\{10, 11, \dots, 14\}$	$\{28, 29, \dots, 38\}$	$\{1, 3, 4\}$
6	$\{10, 11, \dots, 13\}$	$\{28, 29, \dots, 38\}$	$\{2, 5\}$
7	$\{14\}$	$\{32, 33, \dots, 38\}$	$\{2,5\}$
8	$\{42\}$	$\{28, 29, \dots, 38\}$	$\{1, 2, 3, 4, 5\}$
9	$\{43, 44\}$	$\{29, 30, \dots, 38\}$	$\{1\}$
10	$\{45\}$	$\{33, 34, \dots, 38\}$	$\{1\}$
11	$\{43, 44, 45\}$	$\{33, 34, \dots, 38\}$	$\{2, 3, 4, 5\}$
12	$\{13\}$	$\{26\}$	$\{1, 2, 3, 4, 5\}$

Table D1: Combination of sets representing unavailable storage locations

Parameters

Maximum storage height of level z in centimeters. $h_1 = h_2 = 150,\ h_3 = 180,\ h_4 = h_5 = 225$

Maximum pallet capacity of level z. $c_1 = 1{,}728,\ c_2 = c_5 = 1{,}740,\ c_3 = c_4 = 1{,}736$

Appendix E – Supplier-related information

Confidential

Appendix F – Ingredient related information of Company X

Confidential