DESIGNING AN EFFICIENT STORAGE LOCATION ASSIGNMENT APPROACH FOR THROUGH-FLOW WAREHOUSES

by

Fırat Görür

B.S., Industrial Engineering, Istanbul Technical University, 2016

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Industrial Engineering

Boğaziçi University

2022

ACKNOWLEDGEMENTS

Firstly, I would like to thank to my thesis supervisor Assoc. Prof. Gönenç Yücel and my thesis co-supervisor Prof. Mustafa Necati Aras. It would have been impossible to complete this thesis without their guidance in each phase of my research.

I would also like to thank to the jury members Assoc. Prof. Ali Tamer Ünal, Assoc. Prof. Aybek Korugan, and Prof. Mehmet Güray Güler. Their comments enable me to notice the logical errors in the thesis.

Also, I owe thanks to my workmates in Flo Retailing. Their support to me for focusing on the study even in hard times is greatly appreciated.

Lastly but not the least, I thank to my family. They are the ones who make me enjoy everything good, overcome anything bad in my life.

ABSTRACT

DESIGNING AN EFFICIENT STORAGE LOCATION ASSIGNMENT APPROACH FOR THROUGH-FLOW WAREHOUSES

Order placement and retrieval operations in a warehouse with separated incoming and outgoing storage handling areas are analyzed. An approach, which consists of a model and a heuristic algorithm, is devised in order to determine how stock keeping units could be assigned to the storage locations so that the total distance traveled is optimized. All necessary data for the model are generated. Because of the complexity of the problem, a new model with a reformulated objective function, which is derived from number of occurrences in the same SKUs entries and retrievals, is defined to be minimized instead of the SLAP itself. In the objective function; movements between storage handling area of the incoming SKUs and storage locations during the placement operations, movements between the storage handling area of outgoing SKUs and storage locations during the retrieval operations, and movements between the storage locations during the picking operations are included as three components. After modeling, the problem is solved by a heuristic algorithm. Computational results are obtained by 42 different scenarios. The results reveal that good improvements are obtained by the approach, especially when there is no capacity constraint for stock handlers.

ÖZET

AKIŞ TİPİ DEPOLAR İÇİN ETKİN BİR DEPOLAMA LOKASYONU ATAMASI YAKLAŞIMI

Stok toplama ve yerleştirme için iki farklı toplama alanı bulunan depolarda sipariş toplama ve yerleştirme aktiviteleri analiz edilmiştir. İki toplama alanlı bu depolardaki toplam taşıma mesafesini optimize edecek şekilde stokların depolama lokasyonlarına atamasını gerçekleştiren bir yaklaşım; bu yaklaşım kapsamında bir model ve sezgisel çözüm algoritması geliştirilmiştir. Model ve algoritma için gerekli tüm data sanal olarak üretilmiştir. Depolama lokasyonu atama problemlerinin hesaplama karmaşıklığından dolayı, stokların lokasyonlarına yerleştirilmesi ve toplanması sayılarını baz alan bir amaç fonksiyonuna sahip yeni bir model tanımlanmıştır. Modelin amaç fonksiyonunda; gelen ürün toplama alanı ile depolama lokasyonları arasında kat edilen mesafe, giden ürün toplama alanı ile depolama lokasyonları arasında kat edilen mesafe ve depolama lokasyonları arasında kat edilen mesafe bilgisi üç bileşen olarak bulunmaktadır. Modelleme sonrasında sezgisel algoritma aracığıyla 42 test senaryosu için problem çözümlenmiştir. Çözümler incelendiğinde, önerilen yaklaşım ile toplam taşıma mesafesi bağlamında, özellikle stok toplayıcı ve yerleştirici birimlerinde kapasite koşulu bulunmayan test senaryolarında dikkat çekici iyileştirmeler tespit edilmiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

R	Correlation Coefficient Value

LIST OF ACRONYMS/ABBREVIATIONS

ABC Analysis	An Analysis Based on Classification of the Sample Studied
Avg	Average
C++	Name of a Computer Language
C#	Name of a Computer Language
CPLEX	Name of an Optimization Program
E-commerce	Electronic Commerce
GB	Gigabyte
I/O	Incoming/Outgoing
Min	Minimum
NP	Non-deterministic Polynomial Time
OR	Operations Research
PF/PA	Pick Frequency/Part Affinity
PRP	Picker Routing Problem
Qty	Quantity
R	Name of a Computer Language
RAM	Random Access Memory
SKU	Stock Keeping Unit
SL	Storage Location
SLA	Storage Location Assignment
SLAP	Storage Location Assignment Problem
Vs	Versus

1. INTRODUCTION

1.1. Brief Information

Consumption is one of the main activities of humanity. Of course, this comes up with the need for production activities. Therefore, we can say that production is an activity that has been accomplished since the first human was born. Since produced items are not usually consumed as soon as they are produced, storage is also a vital need to keep productions under adverse climatic and other environmental circumstances [1].

With the technological developments and rapid population growth, demand for products and production activities have increased and varied. In today's World, there is a huge amount of production activities. On the other hand, unit production costs are becoming lower thanks to the technological developments, methodology improvements, the economy of scale, and so on. Furthermore, optimization studies also have some influences on these cost reduction activities. Today, one of the main areas of cost reduction is stocking cost. Based on the information provided by managers; 53% of managers find inventory control, 47% of them find picking, and 45% of them find put-away and replenishment as some of the most inefficient production operations [2].

Although warehousing is an important cost center for a supply chain; no stock movements occur in stocking [1]. It is a non-value-added activity for most of the industries, apart from some rare exceptions like the vine industry. Therefore, it is necessary to optimize warehousing activities.

There are lots of studies on warehouse optimization, but in practice, it does not seem possible to remove warehousing activities entirely since these activities also enable to decrease other costs that are occurred in the entire system. Therefore, it is better to begin with stating why we need to stock. The requirement for stocking is derived from the following reasons [2]:

- Uncertainty and fluctuations in demand,
- Optimality of warehousing compared to shipping repetitively,
- Having volume discounts via bulk purchases,
- Cost of the repetitive supplies of products,
- Stoppages in the production activities due to some factors like a crisis, inventory counting, and so on,
- Decreasing the need for adapting production lines frequently,
- Seasonal changes in demand,
- Storage of spare parts to prevent the interruption in production,
- Need for storage of work in progress,
- Adding value by stocking products like wine, antiques, and so on,
- Legal document storage obligations.

In the traditional view, warehouses only function to stock SKUs; however, in today's supply chain view, they support entire sales activities such as product consolidation [1]. Warehouses assume the following roles [2]:

- Storage of raw material,
- Temporary storage during production,
- Storage of finished products,
- Consolidation of items before the delivery,
- Breaking down of items into smaller units of measures,
- Cross-docking,
- Sortation of products in terms of factors like postcodes of delivery location,
- Being fulfilment centers which are designed to meet large amounts of small orders, like e-commerce orders,
- Storage of returning products,
- Storage of public sector items.

The next sections of this chapter provide background information on the warehousing costs, warehouse layout types, storage location-allocation strategies, decision types, and allocation optimization methodologies and algorithm types. Then, some studies in

the literature on the storage location assignment problems are defined and the study of the thesis is positioned.

1.2. Warehousing Costs

There are several kinds of costs to be managed in warehouses. These costs occur during the following operations in warehouses [2]:

- Receiving SKUs,
- Put-away processes,
- Storage,
- Order picking processes,
- Packing SKUs,
- Loading SKUs,
- Stock counting,
- Value-adding services,
- Other activities.

When the distribution of costs occurred in warehouse activities are analyzed, it is seen that the highest percentage is obtained by the order picking processes with approximately 35% [2]. Here, the activities that take order picker's time come into prominence. As pointed out in [3], approximately 50% of order picker's time is consumed by travel activities. It is important since total distances traveled in these travel activities directly affect the time traveled in a warehouse.

It is reasonable to say that layout of warehouses are important factor on the total distances and times traveled. Also, storage location assignments of stock keeping units also affect the times traveled in warehouses, and this directly influences the labor time required. For instance, if SKUs that are frequently picked are placed in storage locations close to the storage handling areas (also known as "depot"), the distances and times traveled to pick these SKUs highly possibly become lower. Here, the important point is that changes in warehouse layouts usually requires investments, whereas better

assignment of SKUs to storage locations does not require additional investments.

In the remainder of this thesis, storage handling areas for both incoming and outgoing SKUs are defined as "handling areas". Storage handling areas for incoming SKUs are defined as "incoming handling areas", and those for outgoing SKUs are defined as "outgoing handling areas".

1.3. Warehousing Layout Types

There are some typical kinds of warehouse layouts due to the specific requirements in warehouses. For instance, if there is not enough place for a warehouse, it may be built as a multi-storey building in a small area, and the layout must be designed suitable to this multi-storey structure. There are two common layout types. They are known as U-flow and through-flow warehouse types.

1.3.1. U-Flow Warehouse

U-flow warehouses are the most common warehouse layout type [2]. In this warehouse type, there is a single handling area in which check-in and delivery operations are performed. A generalized layout for a U-flow warehouse can be seen in Figure 1.1 where the SKUs with higher usage are located in darker shaded areas.

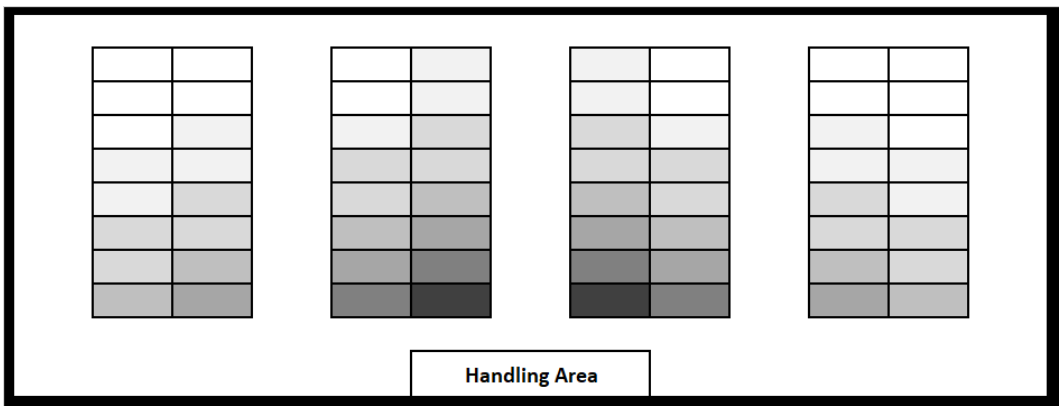

Figure 1.1. Typical U-Flow Warehouse Layout.

1.3.2. Through-Flow Warehouse

In this warehouse type, handling areas for receiving and delivery are located in different places. A typical through-flow warehouse layout may be seen in Figure 1.2.

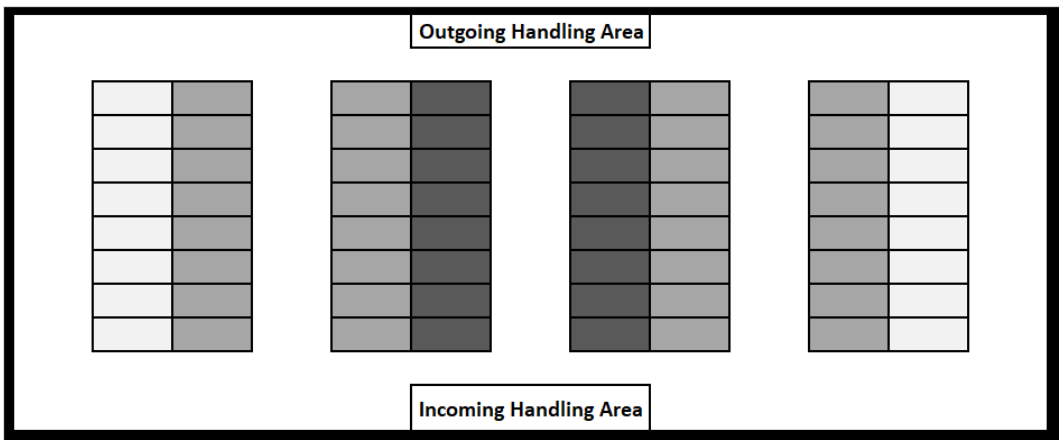

Figure 1.2. Typical Through-Flow Warehouse Layout.

As can be imagined, several routing cycles are required for picks and placements of the SKUs in U-flow and through-flow warehouses. As Bartholdi and Hackman point out in [4], these cycles may be grouped as single-cycles and dual-cycles.

If a forklift goes to stow some amount of an SKU and comes back with empty forks, or a forklift goes to retrieve some amount of an SKU with an empty fork and comes back with the SKU, such a cycle is called a "single-cycle" [4]. In a single-cycle process, the distance traveled for picking or stowing an SKU in a U-flow warehouse is simply twice the distance between the storage area of the SKU and the handling area. This calculation also applies to the single-cycle processes of a picker and a stower in a through-flow warehouse. The only difference is that the distances from the outgoing and incoming handling areas are considered for the picker and the stower, respectively. In single-cycle operations, there are $n!$ possible sequences of routes when n visits are required to stow or retrieve all SKUs in an order and there is only one visit made to each storage location in each of the routes. On the other hand, the total distance traveled for completion of all routes does not change. Examples of single-cycles of a placement and a retrieval operations are illustrated in Figure 1.3. In the figure,

travels with stocks are depicted with solid arrows. On the other hand, travels without stocks are depicted with dashed arrows. In the figure, the movements between the first storage location and the handling area together illustrate an example of a single-cycle placement operation. The movements between the second storage location and the handling area illustrate an example of a single-cycle retrieval operation.

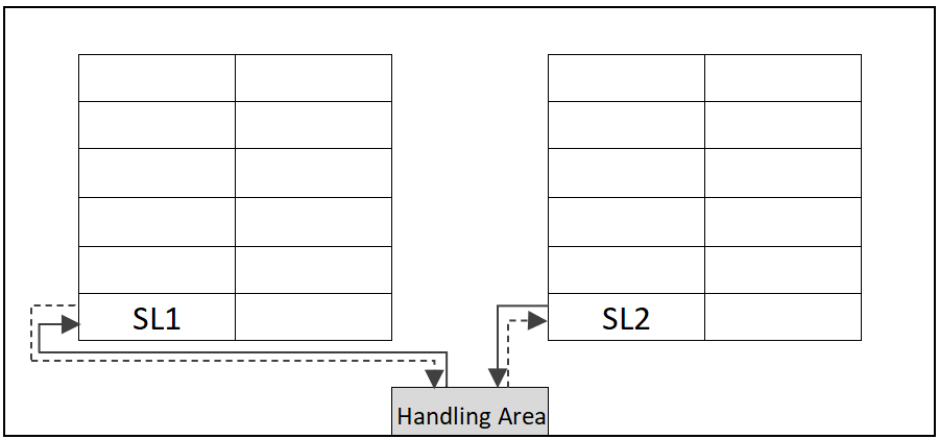

Figure 1.3. Single-Cycle Example.

If a forklift goes to stow a pallet and takes another pallet before coming to the handling area, or vice versa, such a cycle is called a "dual-cycle" [4]. In the dual-cycle operations, there are $(n!)^2$ possible sequences of visits if there are n picking and n stowing visits are required [5]. The dual-cycle examples are illustrated in Figure 1.4. In the figure, travels with stocks are depicted with solid arrows. On the other hand, travels without stocks are depicted with dashed arrows. According to this example, a stock handler picks SKUs from the handling area and stows them into SL1. After the placement of the SKUs, it goes to SL2. Then, it picks SKUs from SL2 and goes to SL3, and stows them there. Finally, the stock handler comes back to the handling area without any SKUs.

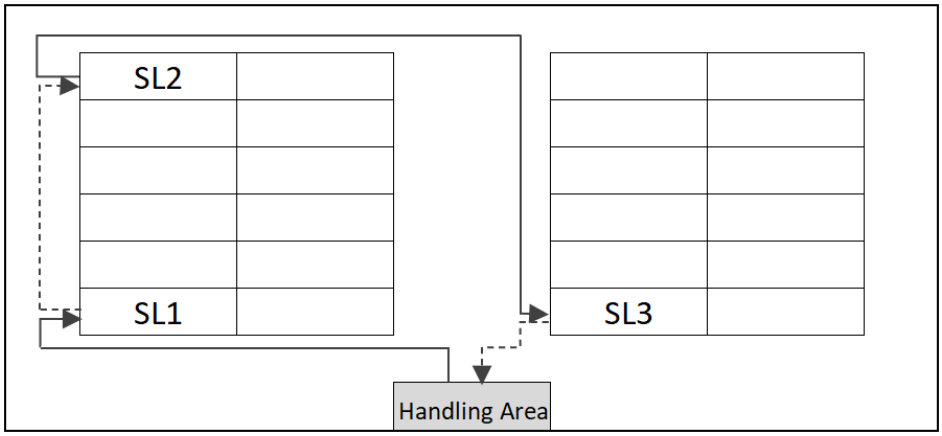

Figure 1.4. Dual-Cycle Example.

In the remainder of the thesis, operators and vehicles that both pick and stow SKUs are defined as the "stock handlers". In addition, the stock handlers are classified as "pickers" and "stowers". Pickers are the stock handlers that only pick SKUs from the storage locations. Stowers are the stock handlers that only place SKUs to the storage locations.

Apart from these two aforementioned cycle types, it is possible to discuss another cycle type for a stock handler with enough carrying capacity to visit multiple storage locations without offloading. It is defined as a multiple-cycle [5]. If a stock handler needs to visit k storage locations from the handling area and then needs to return back after completing all the visits in the picking route without any capacity constraints, there are $k!$ possible routes. If there is a single handling area, there are at most $\frac{(k)!}{2}$ possible total distances traveled. On the other hand, this number is k! for through-flow warehouses. In the remainder of this thesis, multiple-cycles are referred as "multi-cycles".

1.4. Storage Location Assignment Strategies

Storage location assignment is simply the assignment of SKUs and storage locations to each other so that the SKUs are placed to the their assigned storage locations. As pointed out, it affects the total distance and time traveled by stock handlers in a

warehouse. Thus, it is a promising area for optimization studies. Warehouse allocation strategies can be grouped in terms of dedication over time and number of visits to storage locations.

1.4.1. Strategies in Terms of Dedication Over Time

Warehouse allocation strategies may be classified in terms of dedication over time. Bartholdi and Hackman [4] classified them as three types of warehouse allocation strategies:

1.4.1.1. Dedicated Storage. In this strategy, each storage location is assigned to only one SKU, and it does not change over time. In that system, average storage location space utilization is about 50%. Especially when there are fluctuations in demand, these fluctuations can result in insufficient storage capacity problems.

1.4.1.2. Shared Storage. In this strategy, incoming SKUs are assigned to the best available storage location space, and these assignments are dynamic. In that system, storage location space utilization is higher. On the other hand; warehouse management systems are usually required to apply this strategy since the storage location assignments frequently change.

1.4.1.3. Hybrid Storage. In this strategy, SKUs are grouped according to predefined factors such as turnover rates, product types (foods, chemicals, and so on). Each SKU group has a dedicated storage area. On the other hand, the shared storage policy is applied within each of the dedicated individual storage areas.

1.4.2. Strategies in Terms of Number of Visits

Classification of warehouse allocation strategies according to number of visits to storage locations may also be seen in literature like [6, 7]. In that aspect, it is possible to talk about three types of warehouse allocation strategies, which are defined below:

1.4.2.1. Popularity-Based Storage. In this strategy, more frequenly ordered SKUs, namely the popular ones, are located in more accessible storage locations. This decreases the distances traveled especially for single-cycle operations, yet causes crowds in the accessible storage locations [3].

1.4.2.2. Affinity-Based Storage. In this strategy, SKUs that are frequently ordered together, namely the affine ones, are located in the storage locations that are close to each other [3]. When stock handlers operate with dual-cycle and/or multi-cycle, this strategy reduces the distances traveled between storage locations. However, which SKUs are ordered together is a quite dynamic and detailed information. This situation yields the need for a high amount of computational effort.

1.4.2.3. Hybrid Storage. Apart from the popularity-based and affinity-based storage location assignments, hybrid allocation strategies which combine both popularity-based and affinity-based allocation strategies are also applied. For instance, popularity and affinity factors are combined in [6–8].

1.5. Decision Types

Order picking management decisions may be classified as strategic, tactical, and operational decisions [9]. On the other hand, this classification exists not only for order picking decisions but also any kind of other managemental decisions. Gils *et al.* [9] define these decisions as below:

- Decisions taken on using resources efficiently and effectively to gain long-term competitive advantage, like warehouse layout decisions, may be classified as strategic decisions.
- Decisions taken for the medium horizon, like storage location assignment, may be classified as tactical decisions.
- Short-term decisions, like routing decisions, may be classified as operational decisions. These decisions are usually taken on daily operational choices.

1.6. Optimization Methodologies and Algorithm Types

Storage location assignment optimization is one of the main focuses of optimization studies because it is directly related to order picking time and the cost of labor. Storage location assignment problems are done in a way that given key performance indicator's value is optimized [10]. In most of the studies, they are minimization problems.

There are several algorithms to solve the optimization problems. These algorithms may be classified as two main algorithms types [11]:

- Exact Algorithms,
- Heuristic/Metaheuristic Algorithms.

Exact algorithms provide an optimal solution for the optimization problem considered [12]. Some examples of these algorithms are linear programming algorithms, branch-and-bound algorithm, and so on. These algorithms can be utilized for smaller problems. When the problems are complicated, heuristics and metaheuristics are frequently applied [13].

Just like other optimization problems, storage location assignment problems may be optimally solved by exact algorithms. Some of the most frequently used mathematical models for these problems are mixed-integer linear or mixed-integer nonlinear models. Even if the models used are linear, a great deal of computational effort to solve the SLAP models may be required due to the NP complexity of these models [14].

For the complex problems, heuristic and metaheuristic algorithms are usually used. Metaheuristics are built with general-purpose search methods to general problems which are usually independent of specific problems, whereas heuristics are designed for specific problems [13]. Some logical structures are built within heuristic algorithms. Then, these algorithms applied instead of solving an optimization problem with exact algorithms. It is not required to use exact formulation of the problems in heuristic

algorithm. For instance, an exact formulation of the SLAP is used in the heuristic which is built in [15]. On the other hand, SKUs are assigned to storage locations according to a score defined for the SLAP in [7]. When the model for this approach in [7] is analyzed, it is seen that the study is based on a different model with an objective function that reformulates the actual problem. Here, it is reasonable to classify models into actual and reformulated models.

Actual models for SLAPs have direct formulations of the total distances traveled in their objective functions, and include all necessary data and constraints to represent the actual problem. Aras and Görgülü built this model in [15]. On the other hand, reformulated models are basically proxy models which give good solutions for the actual problems, like in [16] and in this thesis.

For SLAPs, heuristics often include stock turnovers or total number of appearances of the SKUs in orders and/or total number of appearances of each pair of SKUs in the same order. Chuang *et al.* [8] define number of appearances of SKUs in orders as the "popularity" of SKUs. Number of visits to storage locations to pick an SKU may also be named as popularity. Apart from the popularities, Smyk [3] defines the pairs of SKUs as "similar" if these pairs frequently appear in the same orders. Kofler *et al.* [7] defined it as "affinity". The terms "popularity" and "affinity" are used in the rest of the thesis.

In the remainder of the chapter, SLAP studies in the literature are mentioned. Then, the study of the thesis and its contributions to literature are pointed out. Finally, contribution of the reviewed literature sources on the study of this thesis is mentioned.

1.7. SLAP Studies in Literature

As Bartholdi and Hackman pointed out in [4]; if material flow in a warehouse is stable and predictable; the SKUs may be dynamically sorted in a list from the most to least turnovers, and storage locations from the least total distances to handling areas to most. Then, assignments are made by pairing them through the lists. It is

a typical greedy assignment algorithm. However, if the warehouse is not at a steady-state, SKUs may be sorted by their departure time. Then, the same remaining steps explained above may be applied.

This greedy algorithm may be seen in many resources, like in [17]. It gives the optimal solution in U-flow warehouses when stock handlers operate with only single-cycles. However, the dual-cycle and multi-cycle are becoming more and more popular today, and this requires advancements in this algorithm.

Zhang [18] built two clustering algorithms based on affinity values and clustered SKUs. Then, he assigned the clusters to storage location clusters, namely storage regions, in terms of popularity. However, he did not consider how close the SKUs within these clusters are placed in the warehouse. This study is conducted on a U-flow warehouse that has a single handling area. He assigned one SKU to only one storage location, and tried to minimize only the total picking distance traveled.

Yang [19] built a reformulated model, and devised a heuristic algorithm similar to a greedy algorithm that is used for the knapsack problem to find a solution close to optimality. Through the algorithm; he sorted the SKUs by their sales profit divided by the space required for these SKUs, and assign the minimum required amount of each SKU to the storage locations. Also, he defined the minimum storage amount required per SKU. After he is finished with the allocation of SKUs until their minimum storage amounts and therefore spaces required; he reordered and allocated the remaining SKUs to the remaining storage locations.

Özyörük and Ak [20] implemented a simpler version of Yang's reformulated model. They determined the priority index of SKUs by using multi-criteria ABC analysis. They grouped the SKUs by their profit and determined coefficients for each SKU group. Then, they solved the reformulated model with an exact algorithm.

Ayhan *et al.* [21] implemented both Yang's model and heuristic algorithm. Moreover, they made an improvement. They also examined the cross profits of SKUs when

the SKUs are located in the adjacent storage locations due to the fact that some SKUs must not be located close to each other. For instance, a dairy product can not be put in a storage location close to the storage locations that contain chemicals.

Aras and Görgülü [15] developed an actual mixed-integer programming model and devised a heuristic algorithm that combines both storage location assignment problems and routing optimization problems, and operates by applying seven move operators. After solving the SLAP for both small and relatively larger instances, they compared the solutions and computational effort of their heuristic with the solutions and computational effort of an exact algorithm which is run on CPLEX. This study is conducted on a U-flow warehouse which has a single handling area. They assigned one SKU to only one storage location. Also, they assigned one storage location to only one SKU. They tried to minimize the total picking distance traveled and did not include distances traveled to stow SKUs in their models.

Wang and Zhang [22] defined a model for the assignment of SKUs and deciding whether storage locations of SKUs need to be reassigned or not at the end of the specified periods. In the model, each storage location assignment is done for each period. Also, a fixed coefficient, which can be considered as a cost, for reassignment is defined. The coefficient for each period does not change with respect to how comprehensive the reassignments are. They optimize the storage location assignments of a whole year by minimizing a function of sum of the total distances traveled and replacement costs. In this model, the total distance traveled is computed assuming that the pickers operate with only single-cycles.

Evinsel [16] studied storage location assignments in a leading durable consumer goods firm. He made an ABC analysis according to how often each SKU is picked in the warehouse. Then, he assigned some coefficients to SKUs according to this classification, and used these coefficients in the objective function of his reformulated model. In his model, he tried to minimize the sum of the coefficients of the SKUs multiplied by the sum of binary decision variables for each possible assignment multiplied by the distances between assigned storage locations and the handling area. It is assumed that

number of the storage locations to be assigned to each SKU is fixed, and does not change with respect to fluctuations demand.

Çolak *et al*. [23] used multi-criteria ABC analysis, and determined priority coefficients of the objective function of their mixed-integer linear programming model. Also, they add constraints to make some individual SKUs are located in adjacent storage locations. Just like the previous study explained above, it is assumed that number of storage locations to be assigned to each SKU is fixed and does not change with respect to fluctuations in demand. The main difference between this study and the previous one is that Çolak *et al*. used multi-criteria ABC analysis, which uses storage life of the each SKU, the amount of the each of the SKUs to be moved, procurement lead time of the each SKU, and unit cost of each SKU as four inputs. They grouped the SKUs according to these four inputs. Then, they defined coefficients for each SKU group, and used these coefficients in their model.

Chuang *et al*. [8] built an approach that includes two integer programming models and a simple heuristic which is based on static historical data. In their first integer programming model, they assigned SKUs to classes. Then, they used these classes like SKUs in their second integer programming model. They multiplied the distances of the storage locations to the handling areas by the order frequencies of assigned SKU classes and minimized their summation in this second model. Finally, they sorted the SKUs in each of the classes in terms of number of occurrences in the pick orders, and assigned them to the storage locations from the closest to the outermost. 65 SKUs are examined in the model. The study is conducted on a U-flow warehouse which has a single I/O point. They assigned one SKU to only one storage location. Also, they assigned one storage location to only one SKU. They tried to minimize the total picking distance traveled.

Liu [24] devised both a reformulated model and a heuristic approach. In this approach, he obtained an initial solution by assigning SKUs in terms of their popularities. Then, he hierarchically clustered SKUs in terms of the affinity measures which are based on the objective function values of their model. Finally, the groups merged

in each clustering step are examined. SKUs of the same groups which are not next to each other are selected hierarchically in terms of popularities and become adjacent. Then, the total distance traveled is calculated and compared with the previous assignment. If there is an improvement obtained in terms of the total distances traveled, this replacement becomes current. Then, this algorithm iteratively continues with each clustering step. This study is conducted on a warehouse which has a single handling area to minimize the total picking distance traveled.

Mirzaei *et al.* [25] made a heuristic algorithm for a part-to-picker warehouse with a single handling area. In this warehouse, SKUs are transported with their assigned storage space by a robot. Each storage space has multiple storage locations in it. In this algorithm, they sorted the SKUs in a list in descending order by the turnover frequencies. The algorithm picks the first SKU in the list and assigns it to the closest storage space. Then, they assign the SKU with the highest affinity between the first assigned SKU. They continue it until no other choices are left. After the assignment of the most popular SKU and its affine SKUs; it repeats the same procedures with the second most popular SKU which is not assigned yet and the second closest storage space. When the assignments are done, they use these assignments as an initial solution to optimize their mathematical model by solving an optimization problem. With that, they compare the performance of the heuristic with a near-optimal result. Here, the distances between the storage locations within the storage spaces are not considered. The focus of this research is to locate affine SKUs in the same storage spaces. It is applicable for part-to-picker warehouses; nonetheless, some additional analysis on the determination of the size of clusters is required for conventional picker-to-part warehouses.

Kofler *et al.* [7] built an approach with a reformulated objective function to minimize the predefined costs of assignments. In their objective function, they simply multiplied part affinity values of SKUs by the distance between their assigned locations and pick frequencies with the distances between assigned locations and the handling area. Here, "part affinity" refers to the frequency that a pair of SKUs is picked in the same retrieval route, and each SKU pair has its corresponding part affinity value. They

called the reformulated objective function value the "PF/PA score". They pointed out that PF/PA scores are easy to be implemented in heuristic algorithms. These algorithms logically assign SKUs frequently picked together to storage locations close to each other, and popular SKUs to storage locations close to the handling area. They achieved to decrease the total picking distance traveled by 7% by using heuristics which considers the PF/PA scores. In the remainder of this thesis, Kofler *et al.*'s approach is stated as "PF/PA approach".

In the literature review above, all warehouses analyzed are U-flow warehouses, except Mirzaei *et al.*'s [25] study. There are not any studies found in the literature review that separated handling areas for incoming SKUs and outgoing SKUs are analyzed. Also, in all of the studies reviewed above, only the total picking distances traveled are considered without considering distances traveled for placement activities. Therefore, to the best of our knowledge, these two aspects in literature are open for contribution. Also, it is seen that the approaches with heuristic algorithms based on reformulated models are started to be used more recently.

1.8. The Study and Its Contribution

In the thesis, storage location assignments in fictional through-flow warehouses, whose stowers and pickers operate with single-cycles and multi-cycles, respectively, are analyzed. Since both single-cycle and multi-cycle operations are analyzed, the study proposes a hybrid allocation strategy which combines popularity and affinity factors.

An approach, which consists of a reformulated model and a heuristic algorithm, is proposed and used to solve the SLAP to minimize the total distance traveled related to the put-away and order picking operations, based on the assumption that the total distance traveled directly reflects the time spent by the stock handlers. Here, the actual problem is not directly implemented in the proposed model because of the complexity of the SLAP. Instead, the proposed model takes a reformulated objective function, which can be used for estimation of the total distances traveled, as the basis. Here, the objective function which is derived from the formulation for PF/PA score is extended

and included in the proposed model's objective function. Therefore, the proposed model can be classified as a reformulated model which indirectly aims to minimize the total distance traveled by minimization of the reformulated objective function. Also, search methods of the proposed heuristic algorithm are built based on the contribution of each SKU on the objective value of the proposed reformulated model.

Kofler *et al.*'s approach in [7] based on the PF/PA scores is extended in two ways. Firstly, Kofler *et al.*'s approach is built for U-flow warehouses. On the other hand, the proposed approach in the thesis is also improved to be suitable for through-flow warehouses. Secondly, Kofler *et al.*'s approach aims to reduce only the total picking distances traveled. However, as pointed out, the main objective of the proposed approach is to reduce the total distances traveled due to both SKU retrieval and stowing activities. By including the two aforementioned ways, lower total distances traveled compared to Kofler *et al.*'s approach and a popularity-based greedy heuristic which is defined by Bartholdi and Hackman in [4] are aimed to be obtained. The mathematical expression of the reformulated models of Kofler *et al.*'s approach and the proposed approach of this thesis are defined in Chapter 2. In the remainder of the thesis, the proposed approach of the thesis is referred the "proposed approach".

The outcome of the proposed approach is an assignment set that does not change with respect to time. Therefore, the approach proposes a dedicated storage location assignment strategy.

The proposed approach focuses on the optimization of the storage location assignment. Therefore, we may state that the solution of the proposed model is a proposed storage location assignment decision which is classified as a tactical decision.

1.9. Contribution of the Literature on the Study of the Thesis

Some of the above-mentioned literature resources have been benefited during the determination of the proposed approach. The benefited areas in this thesis are listed in Table 1.1.

Table 1.1. Contribution of the Literature on the Study of the Thesis.

Resource	**Benefited Areas from the Literature**
[4]	Comparison of approaches in terms of the total distances traveled
[7]	Comparison of approaches in terms of the total distances traveled
[7]	Importance scores used in the greedy sub-algorithm
[7]	Mathematical interpretation of the proposed model
[7]	Calculation of the objective function scores
[16]	Classification of SKUs
[22]	Reassignment of SKUs
[23]	Classification of SKUs
[24]	Initial greedy solution, reverting move operators
[25]	Mathematical interpretation of the proposed model
[25]	Assignment of affine SKUs to closer storage locations

2. PROBLEM DEFINITION

As pointed out before, the main focus of this thesis is the minimization of the total distance traveled due to the SKU placement and retrieval activities in the through-flow warehouses. Since large instances of SLAPs can not be solved to optimality in an acceptable time due to their computational complexity, the SLAP is not optimized by solving an optimization problem. Rather, an approach with a reformulated model and a heuristic algorithm based on this reformulated model, namely the proposed approach is built.

In the reformulated model's objective function, total distances traveled are tried to be estimated. The important point here is that the minimization of the total distances traveled is the main objective for the SLAP in this thesis. In order to determine total distances traveled, picker routing problems also need to be solved. If total distances traveled can be estimated easily and with a good accuracy, then PRPs do not have to be solved as a part of the SLAP. Instead, estimated total distances traveled can be obtained very easily.

At this point, a simplified optimization model, namely the reformulated model, based on the estimated total distances traveled can be solved. Also, a heuristic algorithm based on the estimated total distances traveled can be used to obtain better allocations. As will be mentioned in Chapter 4, a heuristic algorithm is used because it obtains better results in a shorter time compared to optimizing this reformulated model by solving an optimization problem with a large number of SKUs and storage locations.

As mentioned, Kofler *et al.* [7] proposed an objective function which formulates a score, that is named the "PF/PA score". The notation used to formulate the PF/PA score is defined as as can be seen below:

- α = weight parameter of the popularity-related operations
- θ = weight parameter of the affinity-related operations
- I = set of all SKUs; $1 \leq i \leq n$
- K = set of all storage locations; $1 \leq k \leq m$
- p_i = number of visits to retrieve SKU i
- a_{ij} = number of visits to pick SKUs i and j in the same SKU list for picking
- d_{0k} = distance between the handling area and storage location k
- n_i = number of storage locations assigned to SKU i
- d_{kl} = distance between storage locations k and l
- $x_{ik} = 1$ if SKU i is assigned to storage location k, 0 otherwise

The problem in [7] can be modeled as

$$min\ z = \alpha \sum_{k=1}^{m} \sum_{i=1}^{n} \frac{(p_{0i} d_{0k} x_{ik})}{n_i} + \theta \sum_{l>k}^{m} \sum_{k=1}^{m} \sum_{j>i}^{n} \sum_{i=1}^{n} \frac{(a_{ij} d_{kl} x_{ik} x_{jl})}{n_i n_j} \tag{2.1}$$

s.t.

$$\sum_{i=1}^{n} x_{ik} = 1 \qquad k = 1, \ldots, m \tag{2.2}$$

$$\sum_{k=1}^{m} x_{ik} = n_i \qquad i = 1, \ldots, n \tag{2.3}$$

$$x_{ik} \in \{0, 1\} \qquad i = 1, \ldots, n, \quad k = 1, \ldots, m. \tag{2.4}$$

The objective function (2.1) computes the PF/PA score. Constraints (2.2) ensure that a storage location can accommodate only one SKU. Constraints (2.3) guarantee that one SKU is allocated to a predefined number of storage locations. Constraints (2.4) ensure that decision variables are binary variables.

In the remainder of the thesis, the model defined above is called the "PF/PA model". Also, lists of SKUs to be picked in a route are called "picklists". As can be seen, there are no constraints for each of the picklists in the PF/PA model. Therefore, we can say that not only the objective function but also the entire model is reformulated. This also applies to the model of the proposed approach of the thesis. In the remainder

of the thesis, the reformulated model built for the proposed approach is referred the "proposed reformulated model".

PF/PA model is taken as the basis for the proposed reformulated model built in this thesis. However, the proposed reformulated model also includes the SKU entries to a warehouse. Thus, the "popularity" term is redefined as the "incoming popularity", which refers to number of visits to stow an SKU, and "outgoing popularity", which refers to number of visits to pick an SKU. On the other hand, it is assumed that the SKUs are stowed to their assigned storage locations with single-cycles. Note that there are no visits from a storage location to another one considered in single-cycle operations in the thesis. Therefore, affinity is irrelevant for single-cycle operations. Thus, the affinity data that are derived from the SKU entries are not considered in the model.

Also, the proposed reformulated model is built for through-flow warehouses. Since incoming and outgoing SKUs are handled in separated handling areas, the "handling area" term is redefined as the "incoming handling area" and "outgoing handling area". In the proposed reformulated model's objective function, incoming and outgoing popularities are multiplied by the distances of the assigned storage locations from the incoming and outgoing handling areas, respectively. Also, affinities are multiplied by the distances between the assigned storage locations. Then, scores of the incoming popularity, outgoing popularity and affinity-related operations are calculated. The formulation of each of these three component scores is defined in the remainder of this section.

To sum up, the estimated total distances traveled are tried to be minimized in the proposed reformulated model. The total distances traveled are estimated by three components. These three components are named as can be seen below:

- Score of the incoming popularity-related operations,
- Score of the outgoing popularity-related operations,
- Score of the affinity-related operations.

Also, three weight parameters as follows are defined for these components.

- Weight of the incoming popularity-related operations,
- Weight of the outgoing popularity-related operations,
- Weight of the affinity-related operations.

More detailed information on these three weight parameters are given in the remainder of this section. The notation used to formulate the proposed reformulated model and the three component scores is defined as as can be seen below:

- α = weight of the incoming popularity-related operations
- β = weight of the outgoing popularity-related operations
- θ = weight of the affinity-related operations
- I = set of all SKUs; $1 \leq i \leq n$
- K = set of all storage locations; $1 \leq k \leq m$
- p_{0i} = number of visits to retrieve SKU i
- p_{1i} = number of visits to stow SKU i
- a_{ij} = number of visits to pick SKUs i and j in the same picklist
- d_{0k} = distance between incoming handling area and storage location k
- d_{1k} = distance between outgoing handling area and storage location k
- d_{kl} = distance between storage locations k and l
- $x_{ik} = 1$ if SKU i is assigned to storage location k, 0 otherwise

The first component in the objective function of the proposed reformulated model is the score of the incoming popularity-related operations. During the calculation of the score of the incoming popularity-related operations, incoming popularities of SKUs and distances between storage locations and the incoming handling area are considered. The formulation of score of the incoming popularity-related operations is defined as

$$\sum_{k=1}^{m} \sum_{i=1}^{n} \left(p_{0i} d_{0k} x_{ik} \right). \tag{2.5}$$

The second component in the objective function of the proposed reformulated model is the score of the outgoing popularity-related operations. During the calculation of the score of the outgoing popularity-related operations, outgoing popularities of SKUs and distances between storage locations and the outgoing handling area are considered. The formulation of score of the outgoing popularity-related operations is defined as

$$\sum_{k=1}^{m}\sum_{i=1}^{n}\left(p_{1i}d_{1k}x_{ik}\right). \tag{2.6}$$

The third component in the objective function of the proposed reformulated model is the score of the affinity-related operations. During the calculation of the score of the affinity-related operations, affinities of SKUs and distances between storage locations are considered. The formulation of score of the affinity-related operations is defined as

$$\sum_{l>k}^{m}\sum_{k=1}^{m}\sum_{j>i}^{n}\sum_{i=1}^{n}\left(a_{ij}d_{kl}x_{ik}x_{jl}\right). \tag{2.7}$$

The importance of the above-mentioned three components may vary during the estimation of the total distances traveled in different problem instances. Therefore, these components are weighted in the objective function. The mathematical interpretation of the proposed reformulated model is defined as

$$min\ z = \alpha\sum_{k=1}^{m}\sum_{i=1}^{n}\left(p_{0i}d_{0k}x_{ik}\right) + \beta\sum_{k=1}^{m}\sum_{i=1}^{n}\left(p_{1i}d_{1k}x_{ik}\right) + \theta\sum_{l>k}^{m}\sum_{k=1}^{m}\sum_{j>i}^{n}\sum_{i=1}^{n}\left(a_{ij}d_{kl}x_{ik}x_{jl}\right) \tag{2.8}$$

s.t.

$$\sum_{i=1}^{n}x_{ik} = 1 \qquad k = 1,\ldots,m \tag{2.9}$$

$$\sum_{k=1}^{m}x_{ik} = 1 \qquad i = 1,\ldots,n \tag{2.10}$$

$$x_{ik} \in \{0,1\} \qquad i = 1,\ldots,n,\ \ k = 1,\ldots,m. \tag{2.11}$$

Expression (2.8) is the objective function that computes the total score which consists of scores of the incoming popularity, outgoing popularity, and affinity-related operations. Constraints (2.9) ensure that a storage location can accommodate only one SKU. Constraints (2.10) guarantee that one SKU is allocated to only one storage location. Constraints (2.11) ensure that all decision variables are binary variables.

In the model, the weight parameters of the three component scores in the objective function values do not have specified values. Therefore, the determination of these three coefficients is crucial to obtain good solutions. Determination steps of these coefficients are mentioned in Section 3.1.

It should be noted here that the summation of the objective function value and a fixed value, so-called the "intercept", gives the estimated total distances traveled. However, intercept is not included in the proposed reformulated model of this thesis since it does not have an effect on the solution of the problem.

It should also be noted that it is assumed that one SKU is assigned to only one storage location, and a storage location is assigned to only one SKU. Therefore, the popularities and affinities of SKUs are not divided by number of assigned storage locations in the objective function of the proposed reformulated model.

There are some assumptions made for the proposed approach. These assumptions are listed below:

- The total distances traveled directly reflect the warehousing costs related to the put-away and the order picking operations.
- Each of the storage locations is identical in terms of length, width, and height.
- Each SKU may be assigned to only one storage location.
- Each storage location may be assigned to only one SKU.
- The assignment is dedicated and does not change with respect to time.
- Each of the incoming SKUs is placed to storage locations by one single-cycle.
- Outgoing SKUs are retrieved from storage locations by a picker with a predefined

capacity in terms of the total quantities of SKUs that the picker is able to carry at once. This is referred as the "picker capacity" in the thesis. In different SLAP scenarios analyzed in Chapter 4, three different picker capacities, which are 20, 50 and 1000 are defined. One of these three capacities are assigned to each of the SLAP scenarios analyzed. Detailed information on the picker capacities are given in Chapter 4.

- Storage capacity is the maximum SKU quantity that can be located in the SKU's assigned storage location. The storage capacity of each storage location changes in terms of the assigned SKU, and it is uniformly distributed on $(50, 100)$.
- Incoming SKUs are placed to the storage locations as soon as the quantity in the location drops to the amount less than 40% of the storage capacity of the assigned storage location. After the replenishment, the amount of an SKU in the storage location increases to its storage capacity.
- There are 250 product trees. Each product tree refers to a picklist. Therefore, there are 250 picklists.
- The quantities of SKUs in each of the picklists are uniformly distributed on $(1, 10)$.
- Rectilinear distance is used in the warehouse due to the layout and the stock handlers' capabilities.
- Distances traveled occur on only x and y axis.
- Each of the picklists is retrieved independently without order batching.

3. METHODOLOGY

As stated in the previous chapter, the SLAP is not optimized by solving an optimization problem in the thesis. Instead, an approach that consists of a reformulated model and a heuristic algorithm based on this reformulated model is built.

The proposed reformulated model is already defined in the previous chapter. However, weight parameters of the three components of the proposed reformulated model are not defined yet. Therefore, these values need to be determined first in order to estimate the total distances traveled more accurately. Then, the proposed allocation can be determined by the proposed heuristic algorithm which is based on this proposed reformulated model.

The proposed heuristic algorithm has two main phases. These phases may be defined as below:

- Determination of the weight parameters of the three components of the objective function of the proposed reformulated model,
- Determination of the proposed allocation.

3.1. Determination of the Weight Parameters of the Three Components of the Objective Function

As pointed out, the SLAP is not optimized by solving an optimization problem due to the computational complexity. Rather, the total distances traveled are tried to be estimated by three components with a good accuracy instead of using the actual total distances traveled. This way, a good solution to an actual SLAP may be obtained more easily by the optimal solution of the proposed reformulated model. Here, the three weight parameters of their corresponding components need to be determined. In order to come up with these three coefficients which enable us to estimate the total distances traveled accurately, linear regression is used. Since the total distances

traveled are estimated with these three components, total distance traveled is defined as the dependent variable, and the three components are defined as the independent variables.

In order to come up with better prediction of the total distances traveled, different types of instances should be included so that the estimation is as well as possible. The more trials we would have, the better the estimation of the total distances traveled will be. Also, optimum solution for the proposed reformulated model is more likely to be an optimum solution for the actual SLAP when the total distances traveled are estimated with a better accuracy. Therefore, 100 random allocation instances for a given warehouse and SKU set are generated to determine the three weight parameters by using linear regression.

In each of the instances, all SKUs are randomly assigned to the storage locations. Then, an algorithm, which is written in Python to calculate the three component scores, is run to obtain sample data on these 100 random allocation instances. In the remainder of the thesis, this algorithm is referred as the "proposed objective value calculator algorithm". Also, a routing optimization algorithm, which is written in Python and integrated with Google OR Tools, is run to obtain the optimum total distance traveled on each of the 100 random allocation instances. Then, linear regression is applied, and values of the three weight parameters of the components, which make total distances traveled are estimated by the these components with a good accuracy, are determined.

The proposed objective value calculator algorithm takes the distance matrix of the storage locations, the affinity matrix of the SKUs, and each of the 100 random allocation instances as inputs, and calculates scores the of incoming popularity, outgoing popularity, and affinity-related operations for each of these 100 random allocation instances. These data are used as the sample data on the independent variables of the linear regression model.

The routing optimization algorithm, which is written for this thesis in Python, and integrated with the Python extension of Google OR Tools [26], takes the distance

matrix, orders, SKUs, and stock handlers' information and each of the 100 random allocation instances as inputs. Total distances traveled are calculated by this algorithm. After the routing optimization algorithm is run on each of the random allocation instances, 100 different total distances traveled are obtained. These data are used as the sample data on the dependent variable of the linear regression model.

Apart from the two algorithms above, some algorithms are built for the generation of the required input data. Generation processes of distance matrix, affinity matrix, orders, storage locations, SKUs, and stock handlers are defined in Chapter 4.

3.1.1. Routing Optimization Algorithm

The routing optimization algorithm is written in Python, and integrated with Google OR Tools. In the routing optimization algorithm, an initial solution is obtained by a greedy algorithm. Then, better routes are searched by the guided local search. Both the greedy and guided local search algorithms are embedded in Google OR Tools.

3.1.2. Google OR Tools

OR Tools is an open-source optimization software which is developed by Google Inc. It is mainly used for combinatorial problems [27]. It has packages that are compatible with C++, Python, Java, and C# languages [28].

One of the main usage areas of Google OR Tools is the routing optimization problem. There are four main metaheuristics options available in Google OR Tools for these problems [29]:

- Greedy Search,
- Guided Local Search,
- Simulated Annealing,
- Tabu Search.

As stated in [29], guided local search is usually the most efficient search option for vehicle routing problems. Also, it outperforms the local search and simulated annealing [30]. Therefore, this metaheuristic is used in most of the routing problem examples in the tutorials of Google OR Tools. It is also used in the routing optimization algorithm in this thesis.

Guided local search is a local search algorithm basically penalties the local optimal solutions found in iterations. The penalty is added to the objective function and a new iteration starts considering the modified function, namely the augmented objective function [30]. Then, the algorithm repeats itself until a stoppage condition is met. In Google OR tools, this stoppage condition is the duration parameter defined per iteration. This duration is defined as 1 second per picklist. There is no information provided on the optimality gap in Google OR Tools.

The routing optimization algorithm and proposed objective value calculator algorithm are run on each of the 100 random allocation instances. Then, the data obtained by the three components and total distances traveled are imported to R Studio to apply linear regression to determine the weight parameters of the three components so that the total distances traveled are estimated by these components more accurately.

As a side step, objective function value of the proposed reformulated model in each of the 100 random allocation instances is calculated after the weight parameters are determined. Then, the correlation coefficient between these 100 objective values and total distances traveled is calculated to determine how well the objective function of the proposed reformulated model estimates the total distances traveled.

3.2. Determination of the Proposed Allocation

The values of the weight parameters are determined in the previous phase. Then, a good allocation in terms of the objective function values is searched by the allocation improvement algorithm.

There are two sub-algorithms in the allocation improvement algorithm. These sub-algorithms are named as seen below:

- Main allocation improvement algorithm,
- Further improvement algorithm

The first algorithm obtains a good solution with respect to the objective value of the proposed model. Then, the second algorithm takes this solution, and exchanges assigned storage locations of each pair of SKUs to obtain better results. Detailed information is given in the following of this section.

3.2.1. Main Allocation Improvement Algorithm

The main allocation improvement algorithm starts running with grouping SKUs. Here, the SKU groups are determined by scores, namely the "importance scores", of the SKUs. Importance score of an SKU is calculated as

$$t_i = \alpha\, inpop_i + \beta\, outpop_i + \theta \sum_{k \in I,\, k \neq i} affinity_{ik}. \tag{3.1}$$

The notation used in the formula is defined as follows:

- i : SKU index, $i = 1, 2, \ldots, I$
- k : SKU index, $k = 1, 2, \ldots, I$
- I : number of SKUs
- t_i : importance score of SKU i to be picked
- $affinity_{ik}$: number of occurences that SKUs i and l to be ordered together
- $inpop_i$: number of occurences that SKU i to be placed
- $outpop_i$: number of occurences that SKU i to be picked
- α : weight of the incoming popularity-related operations in the objective function of the proposed reformulated model
- β : weight of outgoing popularity-related operations in the objective function of

the proposed reformulated model

- θ : weight of affinity-related operations in the objective function of the proposed reformulated model

When the importance scores of SKUs are calculated, the SKUs are sorted and listed in terms of their corresponding importance scores. Then, the groups are determined in two grouping policies.

In the first policy, the first $\frac{1}{3}$ of the SKUs in the sorted list are assigned to group 1. The next $\frac{1}{3}$ of them are assigned to group 2, and the remaining $\frac{1}{3}$ of the SKUs are assigned to group 3.

Table 3.1. Grouping Policy 1.

Group	**Indice**	**Proportion**
1	0	33.3%
2	1	33.3%
3	2	33.3%

Since there is a strict seperation between the SKUs, some SKUs with similar importance scores may be assigned to different groups. To prevent this strict classification, a second grouping policy is also defined to be able to evaluate these SKUs with similar importance scores during the same iteration. In this second policy, the first $\frac{1}{6}$ of the SKUs in the sorted list are assigned to group 1. Then, the next two $\frac{1}{3}$ of the SKUs are assigned to the group 2 and group 3, respectively. Finally, the remaining $\frac{1}{6}$ of the SKUs are assigned to group 4.

Table 3.2. Grouping Policy 2.

Group	**Indice**	**Proportion**
1	0	16.7%
2	1	33.3%
3	2	33.3%
4	3	16.7%

When the grouping is done, the algorithm constructs an initial solution with a greedy allocation sub-algorithm. Then, the algorithm cancels the storage location assignments of the SKUs of an SKU group for relocation. Then, the greedy allocation sub-algorithm starts operating again for the reassignment of these SKUs to available storage locations whose allocations are currently canceled. These cancellations and reassignments iteratively continue with the next SKU group until the stopping condition is met. More detailed information can be found in the remainder of this section.

There is a shifting rule between the two grouping policies in the main allocation improvement algorithm. If two identical storage location assignments are obtained through the iterations since the last shift in the grouping policies, the grouping policies are shifted. Then, the algorithm continues iterating with the groups of the shifted grouping policy.

These processes of the main allocation improvement algorithm continue until a stopping condition is met. Then, the main allocation improvement algorithm stops operating and the further improvement algorithm starts operating when one of the two conditions below are met:

- The same allocation is obtained for five times,
- The total count of iterations reaches 1000.

The main allocation improvement algorithm can be defined by the pseudo-code in Figure 3.1.

Order SKUs in the unassigned SKUs list;
$G(0) \Leftarrow$ Grouping Policy 1;
$G(1) \Leftarrow$ Grouping Policy 2;
Group SKUS by importance scores into groups with indices 0, 1, 2 in $G(0)$;
Group SKUS by importance scores into groups with indices 0, 1, 2, 3 in $G(1)$;
currentAllocation $\Leftarrow$ **SLA Greedy Sub-Algorithm (no Allocation)**;
bestAllocation $\Leftarrow$ *currentAllocation*;
$i \Leftarrow 0$;
for $j = 0$ to Large Number **do**
  Cancel all assignments of j mod (len($G(i$ mod 2))'th group in $G(i$ mod 2);
  keep other groups' assignments in *currentAllocation*;
  Update *currentAllocation*;
  newAllocation $\Leftarrow$ **SLA Greedy Sub-Algorithm (currentAllocation)**;
  currentAllocation $\Leftarrow$ *newAllocation*;
  Calculate objective function value of the proposed reformulated model;
  if *currentAllocation* gives lower function value than *bestAllocation* **then**
    bestAllocation $\Leftarrow$ *currentAllocation*;
  end if
  increment j;
  if same allocation is obtained for more than once **then**
    increment i;
  end if
  if same allocation is obtained for five times
  or total count of iterations is 1000 **then**
    return *bestAllocation*;
    break;
  end if
end for

Figure 3.1. Main Allocation Improvement Algorithm.

As can be seen, the main allocation improvement algorithm applies another specific sub-algorithm at the two rows of the pseudo-code. In these two rows, the main allocation improvement algorithm calls the sub-algorithm, and use its response to set to a variable. This sub-algorithm is a greedy allocation algorithm. The sub-algorithm takes the current list of the storage location assignments of the SKUs as input parameter from the main allocation improvement algorithm. Then, it assigns the unassigned SKUs and storage locations. When all SKUs and storage locations are assigned, the greedy sub-algorithm returns the complete list of storage location assignments to the main allocation improvement algorithm. This greedy sub-algorithm can be defined by the pseudo-code in Figure 3.2.

Obtain assigned SKUs and storage locations list from *Inputallocation*;
Obtain unassigned SKUs list and storage locations from *Inputallocation*;
Order members of unassigned SKUs list in terms of the importance scores;
Define *currentAllocation* as an empty list;
for each sku $s1$ in unassigned SKUs List **do**
  $totalScore \Leftarrow 0$, $minScore \Leftarrow$ a large number
  for each location $r1$ in unassigned storage locations list **do**
    Temporarily assign $s1$ to $r1$
    $outPop\_s1 \Leftarrow$ outgoing popularity of $s1$. distance of $r1$ from outgoing handling area
    $inPop\_s1 \Leftarrow$ incoming popularity of $s1$. distance of $r1$ from incoming handling area
    for each sku $s2$ in assigned SKUs list which is assigned to $r2$ **do**
      $outPop\_s2 \Leftarrow$ outgoing popularity of $s2$. distance of $r2$ from outgoing handling area
      $inPop\_s2 \Leftarrow$ incoming popularity of $s2$. distance of $r2$ from incoming handling area
      $affinity\_s1s2 \Leftarrow$ affinity of $s1$ and $s2$. distance between $r1$ and $r2$
      $wOutPop \Leftarrow (outPop\_s1 + outPop\_s2)$. weight of the outgoing popularity-related operations
      $wInPop \Leftarrow (inPop\_s1 + inPop\_s2)$. weight of the incoming popularity-related operations
      $wAff \Leftarrow affinity\_s1s2$. weight of the affinity-related operations
      $totalScore \Leftarrow totalScore + wOutPop + wInPop + wAff$
    end for
    if $totalScore < minScore$ **then**
      Update candidate assignment and *minScore*
    end if
  end for

Figure 3.2. SLA Greedy Sub-Algorithm.

```
    Insert the candidate assignment to currentAllocation
    Insert the sku s1 to Assigned SKUs List
    if all SKUs are assigned to storage locations then
      break;
    end if
  end for
  return currentAllocation;
```

Figure 3.2. SLA Greedy Sub-Algorithm (cont.).

3.2.2. Further Improvement Algorithm

When the main allocation improvement algorithm is done, further improvement algorithm starts running. In the further improvement algorithm, each pair of SKUs are selected and their assigned locations are swapped. Then, the objective function value of the reformulated model is recalculated. If the objective new function value is less than the minimum objective function value obtained until the previous iteration, this change becomes permanent. The stopping condition here is that no better allocation is obtained when each pair of SKUs are evaluated. This algorithm can be defined by the pseudo-code in Figure 3.3.

Require: allocation to be improved
Define the allocation as the *candidateAllocation*
Calculate the objective function value of the candidate allocation as *minimumAllocationScore*
for each sku $s1$ assigned to the storage location $r1$ **do**
for each sku $s2$ assigned to the storage location $r2$ **do**
Temporarily assign $s1$ to $r2$
Temporarily assign $s2$ to $r1$
Define new allocation as *currentAllocation*
Calculate objective function value as *currentAllocationScore*
if *currentAllocationScore* $<$ *minimumAllocationScore* **then**
candidateAllocation $\Leftarrow$ *currentAllocation*
minimumAllocationScore $\Leftarrow$ *currentAllocationScore*
end if
end for
end for
bestAllocation $\Leftarrow$ *candidateAllocation*
if a better allocation is found **then**
FurtherImprovement (bestAllocation);
else
return *bestAllocation*, *bestAllocationScore*
end if

Figure 3.3. FurtherImprovement Algorithm.

After the proposed algorithm is run, the proposed storage location assignments set is obtained for the scenario analyzed. The flow chart of all the steps so far may be seen in Figure 3.4.

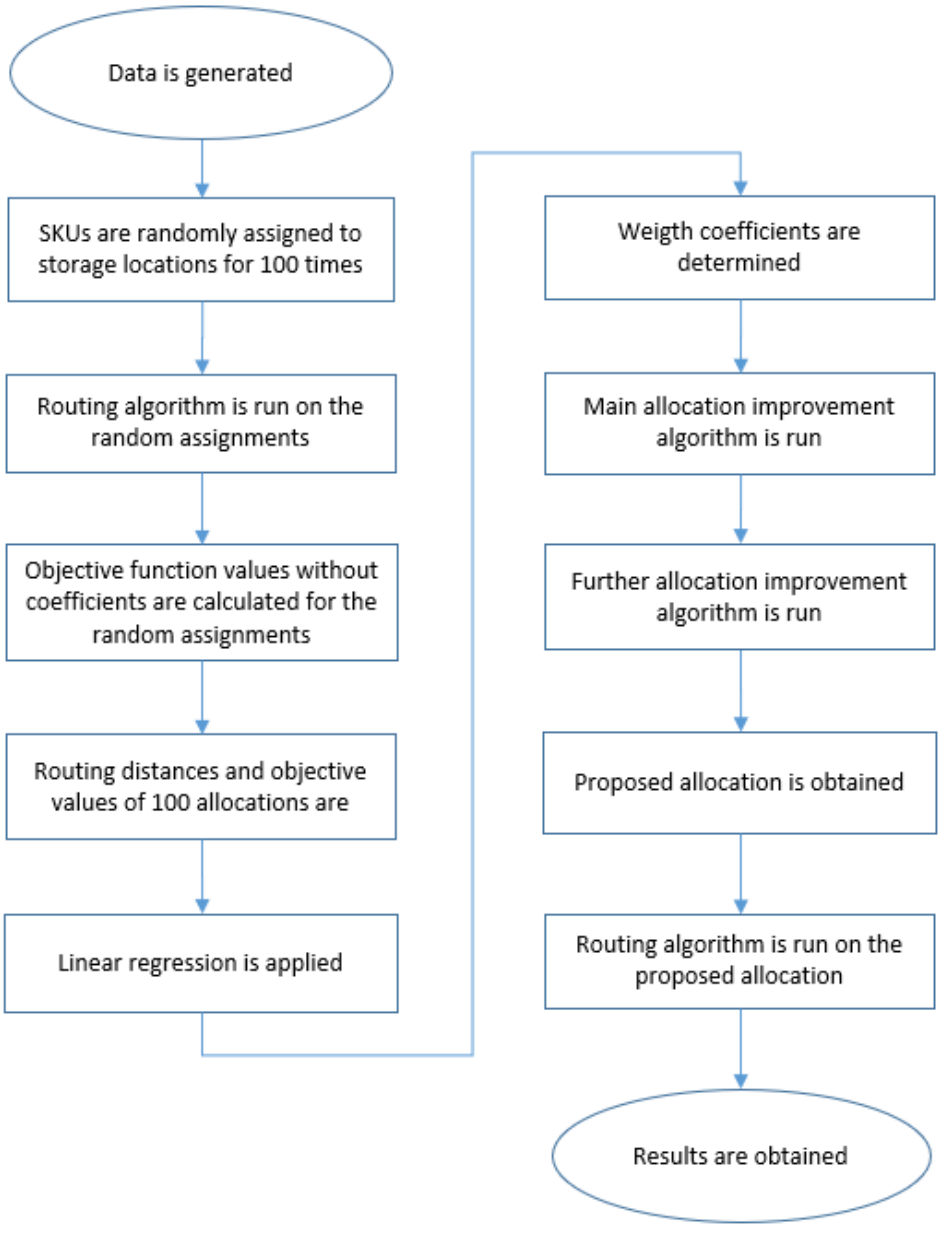

Figure 3.4. Flow Chart of the Methodology.

When the proposed storage location assignments set is obtained, the total distance traveled obtained by this proposed allocation is compared with average of the 100 total distances traveled, minimum of the 100 total distances traveled, the total distance traveled obtained by the PF/PA approach, and the total distance traveled obtained

by the solutions of Bartholdi and Hackman's popularity-based greedy heuristic. Doing this, whether lower total distances traveled are obtained by the proposed approach is analyzed.

3.3. PF/PA Approach's Proposed Allocation Policy

The proposed approach is based on Kofler *et al.*'s [7] approach where the PF/PA score is considered. In addition to this, there are two main improvements on the proposed approach. First of all, the PF/PA approach is based on the total picking distances traveled while the proposed approach is based on the total distances traveled. Therefore, placement distances are also included in the proposed reformulated model of the thesis. Then, the PF/PA approach is more applicable for the U-flow warehouses, whereas the proposed approach can be used for both U-flow and through-flow warehouses.

Since the proposed approach is the extension of the PF/PA approach in two above-mentioned aspects, the improvement on the total distances traveled obtained by the proposed approach compared to the PF/PA approach under these aspects needs to be evaluated. Also, whether the proposed approach is more applicable for through-flow warehouses needs to be proven. However, the only information about the solution algorithm in [7] is that neighborhood search and simulated annealing algorithms are used on the model with different combinations of α and θ values. Therefore, the solution algorithm built for the proposed reformulated model is customized in order to compare the PF/PA approach with the proposed approach. Then, this customized algorithm is implemented to the PF/PA model.

As mentioned in Chapter 2, the PF/PA approach considers only the picking operations during the calculation of the scores. Therefore, the customized algorithm for the PF/PA model is based on the picking routes between storage locations and the outgoing handling area, and between storage locations.

There are five customizations on the proposed algorithm in order to make it applicable for the PF/PA model. First of all, the total picking distance traveled is set as the dependent variable instead of the total distance traveled in the linear regression mentioned in Section 3.1. Secondly, incoming popularity is excluded from the independent variables of the linear regression. Thirdly, incoming handling area is excluded from the solution algorithm since PF/PA approach only concerns with picking activities. Then, the incoming popularity is excluded from the objective function value calculator algorithm. This way, the objective value calculator algorithm calculates the PF/PA score. Finally, the incoming popularity is excluded from the allocation improvement algorithm. To do so, the formulation of the importance score of an SKU and the greedy sub-algorithm is slightly customized. The customized importance score of an SKU is calculated as

$$t_i = weightoutpop\, outpop_i + weightaff \sum_{k \in I,\, k \neq i} affinity_{ik}. \tag{3.2}$$

The notation used in the formula is defined as follows:

- i : SKU index, $i = 1, 2, \ldots, I$
- k : SKU index, $k = 1, 2, \ldots, I$
- I : number of SKUs
- t_i : importance score of SKU i to be picked
- $affinity_{ik}$: number of occurences that SKUs i and k to be ordered together
- $outpop_i$: number of occurences that SKU i to be picked
- $weigthoutpop$: weight of outgoing popularity-related operations
- $weigthaff$: weight of affinity-related operations

The pseudo-code of the greedy sub-algorithm for the PF/PA approach is defined in Figure 3.5.

Obtain assigned SKUs and storage locations list from *Inputallocation*;
Obtain unassigned SKUs list and storage locations from *Inputallocation*;
Order members of unassigned SKUs list in terms of the importance scores;
Define *currentAllocation* as an empty list;
Assign the phantom stocks to incoming and outgoing handling areas;
for each sku $s1$ in unassigned SKUs List **do**
  $totalScore \Leftarrow 0$, $minScore \Leftarrow$ a large number
  for each location $r1$ in unassigned storage locations list **do**
    Temporarily assign $s1$ to $r1$
    $outPop\_s1 \Leftarrow$ outgoing popularity of $s1$. distance of $r1$ from outHandArea
    for each sku $s2$ in assigned SKUs list which is assigned to $r2$ **do**
      $outPop\_s2 \Leftarrow$ outgoing popularity of $s2$. distance of $r2$ from outHandArea
      $affinity\_s1s2 \Leftarrow$ affinity of $s1$ and $s2$. distance between $r1$ and $r2$
      $wOutPop \Leftarrow (outPop\_s1 + outPop\_s2)$. weight of the outgoing popularity-related operations
      $wAff \Leftarrow affinity\_s1s2$. weight of the affinity-related operations
      $totalScore \Leftarrow totalScore + wOutPop + wAff$
    end for
    if $totalScore < minScore$ **then**
      Update candidate assignment and $minScore$
    end if
  end for
  Insert the candidate assignment to *currentAllocation*
  Insert the sku $s1$ to Assigned SKUs List
  if all SKUs are assigned to storage locations **then**
    break;
  end if
end for
return currentAllocation;

Figure 3.5. SLA Greedy Sub-Algorithm.

3.4. Greedy Heuristic's Proposed Allocation Policy

In addition to PF/PA approach, the allocation solutions obtained by the proposed approach is also compared with the solutions obtained by a greedy algorithm. Here, Bartholdi and Hackman's [4] approach is taken as basis.

As pointed out, only the outgoing popularities of SKUs are included in the greedy algorithm which is mentioned by Bartholdi and Hackman. In this algorithm, SKUs are sorted in a list by outgoing popularities from the highest popularity to the least. Then, storage locations are sorted in another list by distances from the outgoing handling area from closest to farthest. Then, SKUs and storage locations with the same indices in their sorted lists are assigned to each other. The pseudo-code of this greedy algorithm is defined in Figure 3.6.

```
Obtain unassigned storage locations list as a list of all storage locations;
Order members of unassigned SKUs list in terms of their outgoing popularity values;
Order members of unassigned storage locations list in terms of their distances from the outgoing handling area;
Define currentAllocation as an empty list;
for each sku s1 in unassigned SKUs List do
   r1 ⇐ the closest storage location to outgoing handling area in unassigned storage locations list;
   assign s1 to r1;
   add the assignment to currentAllocation;
   remove s1 from unassigned storage locations list;
end for
return currentAllocation;
```

Figure 3.6. SLA Greedy Algorithm.

4. EVALUATION OF THE PROPOSED APPROACH

4.1. Test Dataset

All necessary data for evaluation of the proposed approach are generated by algorithms coded in Python. The data consist of storage locations, SKUs and product trees, warehouse layout, picker, stower, and SKUs' movement data. The distance matrix and the affinity matrix are created by using the data generated.

4.1.1. Storage Locations

A warehouse with one incoming handling area, one outgoing handling area, and identical storage locations is defined. Note that the incoming handling area is the location where the entering SKUs are gathered before stowers take these SKUs for placement. Outgoing handling areas are the locations where the SKUs that are picked from the storage locations are gathered for being sent from the warehouse. The handling areas are placed in the opposite parts of the through-flow warehouse which may be seen especially in the production facilities. A storage location can accommodate only one SKU assigned. The width and depth of the storage locations are defined as 1.5 distance units.

4.1.2. SKUs and Product Trees

There is a predefined number of SKUs generated. This information is given in each of the cases in the next section of this chapter. Each of the SKUs can be assigned to only one storage location. Since the storage locations are identical, the storage capacities of the storage locations are determined by the assigned SKUs. The storage capacities of the storage locations are uniformly distributed on the $(50, 100)$ interval.

250 product trees, which may be seen as the bill of materials in the production facilities, are derived from these SKUs. The quantity of each SKU in a product tree is

uniformly drawn between 1 and 10. Demanded quantities for each of the product trees are uniformly distributed on the $(1, 100)$ interval per simulation run. Each of these product trees is refers to a picklist in this paper.

Picklists are basically list of SKUs to be picked in a route. Since there is no batching in picklists, the assigned storage locations of the SKUs of each picklist are the locations to visit in an individual route. In other words, each of the picklists are a seperate routing optimization problem. Since there are 250 product trees defined, there are 250 kinds of picklists.

4.1.3. Warehouse Layout

The generated warehouses have rectangular shapes. There are vertical and horizontal corridors with 3 distance unit width. The lengths of the corridors are calculated by the storage location width and length and the corridor width. The placements of the storage locations within warehouses are characterized by the combinations of number of segments, namely the x-y-z-t-n format. For instance, the sample warehouse which is drawn in Figures 4.1, 4.2, and 4.3 is a 2-4-4-2-2 warehouse. The indices in terms of x, y, z, t, and n are also illustrated in these figures.

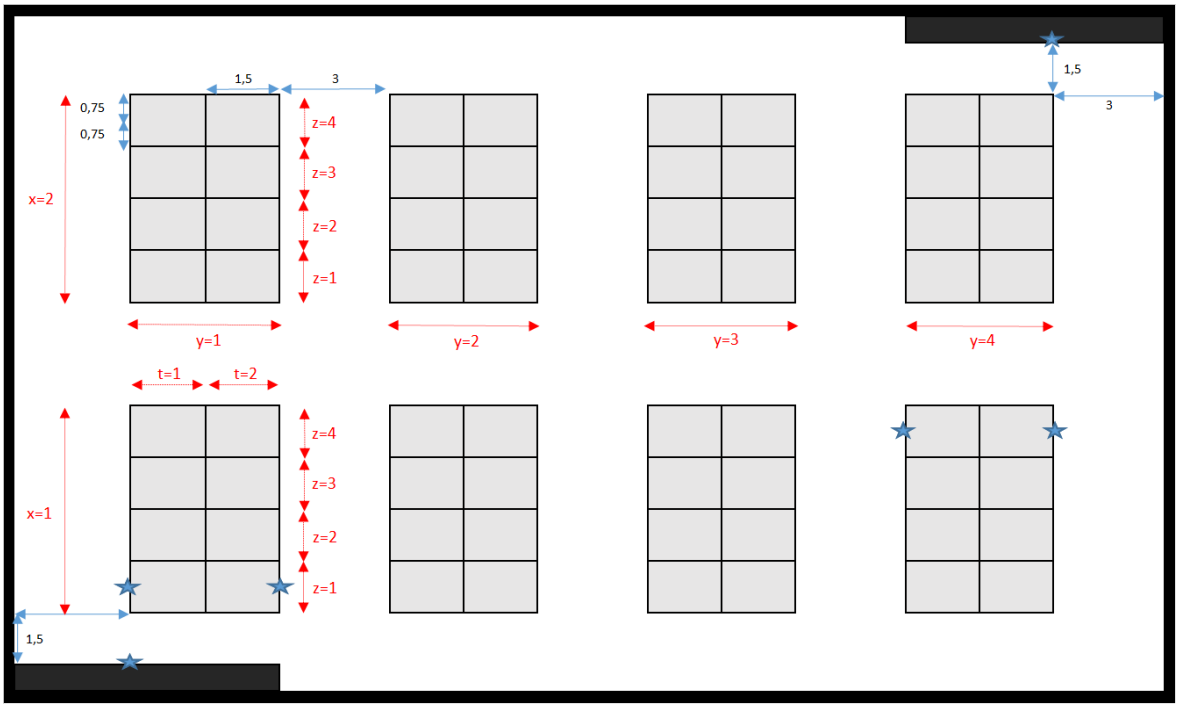

Figure 4.1. Top View for the Sample Layout.

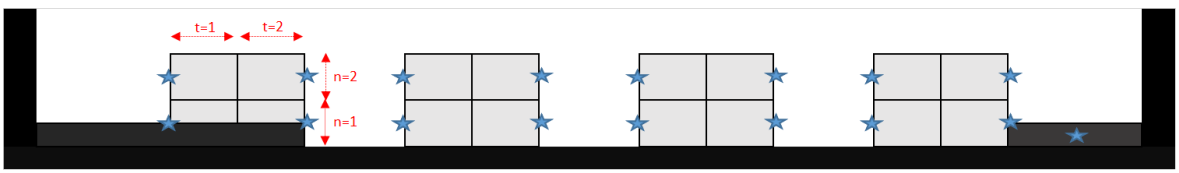

Figure 4.2. Front View for the Sample Layout.

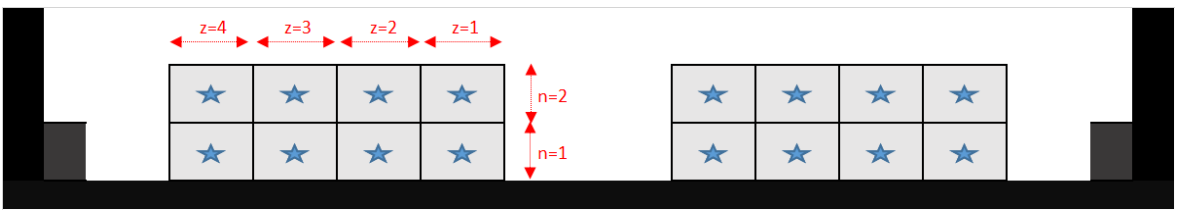

Figure 4.3. Side View for the Sample Layout.

4.1.4. Picker

There is one picker with a parametric capacity in terms of the maximum number of SKUs that can be carried at once. As mentioned, this capacity is defined as "picker capacity" in this thesis. Volumes of SKUs are not considered in determination of the picker capacity. In different cases, picker capacity is taken as 20, 50, and 2000, which refers to no picker capacity constraint without loss of generality. Routes for retrieval of SKUs are determined in such ways that the picker capacity is not exceeded. When the total amount of SKUs to be picked in a picklist is more than the picker capacity, route for retrieval of a picklist may consist of multiple sub-routes.

4.1.5. Stower

There is one stower generated. The stower is assumed to place the SKUs with single-cycle operations.

4.1.6. SKUs' Movement Data

As mentioned before, picklists are derived from the product trees. On the other hand, each of the SKUs is placed as soon as its quantity in its storage location falls below 40% of the storage capacity. After the placement of an SKU, the amount of it reaches the storage capacity in its storage location. Lists of SKUs for placements are generated by considering some initial quantities of the SKUs which are uniformly drawn between 1 and the storage capacity of the assigned location.

After all the above-mentioned data are generated, the three main inputs necessary for the routing optimization and storage location assignment are generated. These inputs are the distance matrix, affinity matrix, and the 100 random allocation instances.

4.1.7. Distance Matrix

Distances between storage locations and the handling areas are calculated with respect to the shortest Manhattan travel distances between the middle points of the front parts of storage locations where placements and picks occur. These shortest distances are calculated by an algorithm coded in Python with respect to only the x and y axis. According to the calculations, a symmetric distance matrix is generated, and it is used in the routing optimization and the allocation improvement algorithms. Also, it is used to solve some small instances by CPLEX in the beginning of Chapter 2.

4.1.8. Affinity Matrix

The occurrences of each pair of SKUs in the same picklists, which are defined as affinities, are calculated and placed into a matrix, namely the "affinity matrix", whose columns and rows refer to the SKUs. There are two virtual SKUs, which are defined as "phantom SKUs" in the thesis, generated to calculate the incoming and outgoing SKU popularities. One of them is virtually added to each picklist so that the affinity of an SKU with this one refers to the outgoing popularity. This SKU is assigned to a phantom storage location which refers to the outgoing handling area. The second phantom SKU is used to calculate the incoming popularities of the SKUs. The affinity of an SKU with this one refers to number of placements of this SKU. This phantom SKU is assigned to another phantom storage location which refers to the incoming handling area. The affinity values between the two phantom SKUs are assumed to be zero. After the symmetric matrix is generated, it is included in the allocation improvement algorithm and the algorithm that is integrated with CPLEX and optimizes the reformulated model by solving an optimization problem.

4.1.9. Random Allocation Instances

100 random allocation instances of SKUs to storage locations are generated in order to run the routing optimization algorithm and the proposed objective value

calculator algorithm on each of these instances to obtain the total distances traveled and the three components of the proposed reformulated model's objective function. As mentioned in the previous chapter, the results obtained by these two algorithms are used as 100 samples in the linear regression model in order to determine the three weight parameters of the three components of the proposed reformulated model's objective function which gives a higher correlation between the objective function values and the total distances traveled.

4.2. Performance Analysis of the Allocation Improvement Algorithm

As can be seen from the mathematical interpretation of the proposed reformulated model in Chapter 2, the objective function has quadratic terms in one of the three components. The decision variables are multiplied by each other in the formula of the score of the affinity related operations. Since the model is not linear, linear optimization methods can not usually give near-optimal solutions in an acceptable time period.

To determine whether the proposed reformulated model can be optimized by solving an optimization problem in an acceptable time period, five test scenarios are defined. The purpose here is to determine whether the allocation improvement algorithm, which is specifically built for the proposed reformulated model, gives better results than optimizing the proposed reformulated model by solving an optimization problem. Since determination of the three weight parameters are not of interest here, these coefficients are taken as 1. Then, the proposed reformulated model is solved for each of the cases by one of the state-of-art optimization tools, CPLEX. Since CPLEX is compatible with Python, the Python extension of CPLEX is used to solve the optimization problem.

The results for the proposed reformulated model, which are obtained by the allocation improvement algorithm and CPLEX, are defined in Table 4.1. It should be noted here that the time consumed for determination of the weight parameters of the three components of the objective function of the proposed reformulated model is not included in the run times given in Table 4.1.

Table 4.1. CPLEX Results.

Number of SKUs	Number of Decision Variables	Number of Quadratic Terms	Run Time		Results		
			Cplex	Proposed Algorithm	Cplex	Proposed Algorithm	Improvement
18	324	104976	1 day	2 seconds	130749.8	130799.3	0.0%
26	676	456976	1 day	3 seconds	294603.8	288599.3	2.0%
42	1764	3111696	1 day	5 seconds	448902.0	444231.0	1.0%
50	2500	6250000	1 day	7 seconds	438259.5	396331.5	9.6%
66	4356	18974736	1 day	13 seconds	857841.0	717901.5	16.3%

As can be seen from the table, for larger instances, the allocation improvement algorithm gives better results in a shorter time period than optimizing the proposed reformulated model by solving an optimization problem in CPLEX. Therefore, the allocation improvement algorithm, which is specifically built for the proposed reformulated model, is used in the thesis.

4.3. Computational Results

In the thesis, 42 different experimental scenarios are defined to obtain computational results for different cases. Each of these 42 experimental scenarios represents an individual SLAP. These scenarios are defined in four main dimensions which are listed below:

- Number of SKUs and storage locations,
- The average number of SKUs in the picklists,
- Warehouse layout,
- Picker capacity.

Each of these scenarios is analyzed in terms of the correlation between the 100 total distances traveled and the objective function values of the proposed reformulated model. Also, each of these scenarios is analyzed in terms of the improvements in the total distances traveled.

As mentioned in Section 3.2, the correlation of each of the scenarios is calculated by the correlation coefficient between the total distances traveled of the 100 random allocation instances of the scenario and corresponding objective values of the proposed reformulated model. The information on the correlations are important since higher correlation comes up with a better estimation of the total distances traveled by the objective function of the proposed reformulated model.

The improvements obtained by the proposed approach in terms of the total distances traveled are determined by comparing them with the average of the 100 total distances traveled, minimum of the 100 total distances traveled, the total distances traveled obtained by the PF/PA approach, and the total distances traveled obtained by the solutions of Bartholdi and Hackman's popularity-based greedy heuristic. The results are given in the Section 4.3.1.

The proposed heuristic algorithm is run on a computer with a 64 bit Intel Core i7 processor and 12 GB RAM. The whole processes of the proposed algorithm from the data generation to the determination of the total distance traveled for the proposed allocation takes approximately 7–8 hours. For example, in one of the cases analyzed in the remainder of this section, the whole process takes 26894.2 seconds, which refers to 7.47 hours. More detailed information may be seen in Table 4.2. Note that most of the time is consumed by the routing optimization algorithm that is integrated with Google OR Tools.

Table 4.2. Runtime for the Main Processes of the the Proposed Algorithm.

Processes	**Run Time (seconds)**
Generation of Data and Matrices	61.6
Routing Optimization on 100 Random Instances	26430.0
Allocation Improvement	136.8
Routing on the Proposed Allocation	252.3
Result Evaluation	13.5

Since there are lots of code lines written for the proposed algorithm, whether the algorithm operates correctly needs to be analyzed. To do that, results are analyzed on a smaller model which is defined for testing purposes. Detailed information on the verification can be found in Appendix A.

As previously stated, there are 42 scenarios defined. The experimental setups for these scenarios are listed in Table 4.3.

Table 4.3. The Scenarios Studied.

Scenario	**Num of SKU**	**Avg SKUs per pick**	**Layout**	**Picker Capacity**
1	36	8.45	3-4-3-1-1	5000
2	36	12.18	3-4-3-1-1	5000
3	36	17.41	3-4-3-1-1	5000
4	36	22.97	3-4-3-1-1	5000
5	72	17.1	2-6-6-1-1	5000
6	72	17.1	2-3-3-2-2	5000
7	72	17.1	1-3-12-2-1	5000
8	72	17.1	1-1-36-2-1	5000
9	72	13.2	2-3-3-2-2	5000
10	72	13.2	2-6-6-1-1	5000
11	72	13.2	1-3-12-2-1	5000
12	72	13.2	1-1-36-2-1	5000
13	128	13.7	2-4-4-2-2	5000
14	128	13.7	2-8-2-2-2	5000
15	128	13.7	2-2-8-2-2	5000
16	128	13.7	1-8-8-2-1	5000
17	128	13.7	4-4-4-2-1	5000
18	128	17.4	2-4-4-2-2	5000
19	128	17.4	2-8-2-2-2	5000
20	128	17.4	2-2-8-2-2	5000
21	128	17.4	1-8-8-2-1	5000

Table 4.3. The Scenarios Studied (cont.).

22	128	17.4	4-4-4-2-1	5000
23	128	13.7	2-4-4-2-2	50
24	128	13.7	2-8-2-2-2	50
25	128	13.7	2-2-8-2-2	50
26	128	13.7	1-8-8-2-1	50
27	128	13.7	4-4-4-2-1	50
28	128	13.7	2-4-4-2-2	20
29	128	13.7	2-8-2-2-2	20
30	128	13.7	2-2-8-2-2	20
31	128	13.7	1-8-8-2-1	20
32	128	13.7	4-4-4-2-1	20
33	128	17.4	2-4-4-2-2	50
34	128	17.4	2-8-2-2-2	50
35	128	17.4	2-2-8-2-2	50
36	128	17.4	1-8-8-2-1	50
37	128	17.4	4-4-4-2-1	50
38	128	17.4	2-4-4-2-2	20
39	128	17.4	2-8-2-2-2	20
40	128	17.4	2-2-8-2-2	20
41	128	17.4	1-8-8-2-1	20
42	128	17.4	4-4-4-2-1	20

4.3.1. The Correlation Between the Total Distances Traveled and the Objective Function Values of the Proposed Reformulated Model

The correlation between the objective function values of the proposed reformulated model and the total distances traveled is important since improvements on the objective function values that are more correlated with the actual problem are more likely to improve the real objectives. Therefore, correlation coefficient between the

total distances traveled of the 100 random allocation instances of each scenario and corresponding 100 objective function values of the proposed reformulated model is calculated in order to determine how well the objective function of the proposed reformulated model estimates the total distances traveled. The correlation coefficients of the 42 scenarios are listed in Table 4.4.

Table 4.4. The Correlation Results of the Scenarios Studied.

Scenario	**Correlation Coefficient (R)**	**R^2**
1	70.3%	49.4%
2	60.8%	36.9%
3	59.0%	34.9%
4	82.4%	67.9%
5	68.9%	47.5%
6	72.6%	52.7%
7	88.1%	77.7%
8	100.0%	100.0%
9	75.2%	56.5%
10	78.6%	61.8%
11	85.2%	72.6%
12	100.0%	100.0%
13	71.2%	50.7%
14	88.1%	77.7%
15	72.9%	53.1%
16	81.1%	65.7%
17	69.8%	48.7%
18	85.1%	72.4%
19	92.7%	86.0%
20	84.2%	70.9%
21	85.8%	73.6%
22	76.0%	57.7%
23	58.2%	33.8%

Table 4.4. The Correlation Results of the Scenarios Studied (cont.).

24	61.8%	38.2%
25	56.7%	32.1%
26	52.1%	27.2%
27	43.5%	19.0%
28	86.1%	74.2%
29	85.0%	72.3%
30	87.3%	76.2%
31	85.3%	72.8%
32	86.3%	74.5%
33	70.6%	49.8%
34	71.5%	51.1%
35	75.9%	57.5%
36	54.5%	29.7%
37	65.1%	42.4%
38	89.5%	80.2%
39	87.6%	76.7%
40	84.8%	71.9%
41	85.0%	72.2%
42	85.6%	73.2%

As can be seen especially in some of the 42 scenarios above, the proposed reformulated model's objective function values and the total distances traveled are highly correlated for the 100 random allocation instances. On the other hand, they are less correlated in some cases. Therefore, the reasons for these changes need to be analyzed in order to figure out in which cases the proposed approach gives better results.

According to the results, the proposed approach is quite applicable for most of the incapacitated scenarios for through-flow warehouses where stowers operate with single-cycles and pickers operate with multi-cycles. Here, incapacitated scenarios refer to the scenarios with no picker capacity constraints, and capacitated scenarios refer to

the scenarios with picker capacity constraints. On the other hand, the results show that the correlation values first decrease with the decreasing picker capacity. To visualize, the correlation values of the cases with 128 SKUs, 250 product trees, and an average of 12.7 SKU items in the picklists except the phantom SKUs are illustrated in Figure 4.4. The main reason here is that the information on the affinities and outgoing popularities alter because of the sub-routes. After some point, the correlation values increase with the decreasing picker capacity. The main reason is that the picking activities start to be with single-cyles. Moreover, the total distances traveled between the storage locations and the outgoing handling area increases. As a result, the weight of the outgoing popularity-related operations increases in the proposed reformulated model. If we would have considered only the picking activities and there would have been only single-cycle picking activities; there would be a perfect fit between the total distances traveled and the objective function values of the proposed reformulated model, like how it happens between the scores of the incoming popularity-related operations of the objective function and the total placement distances.

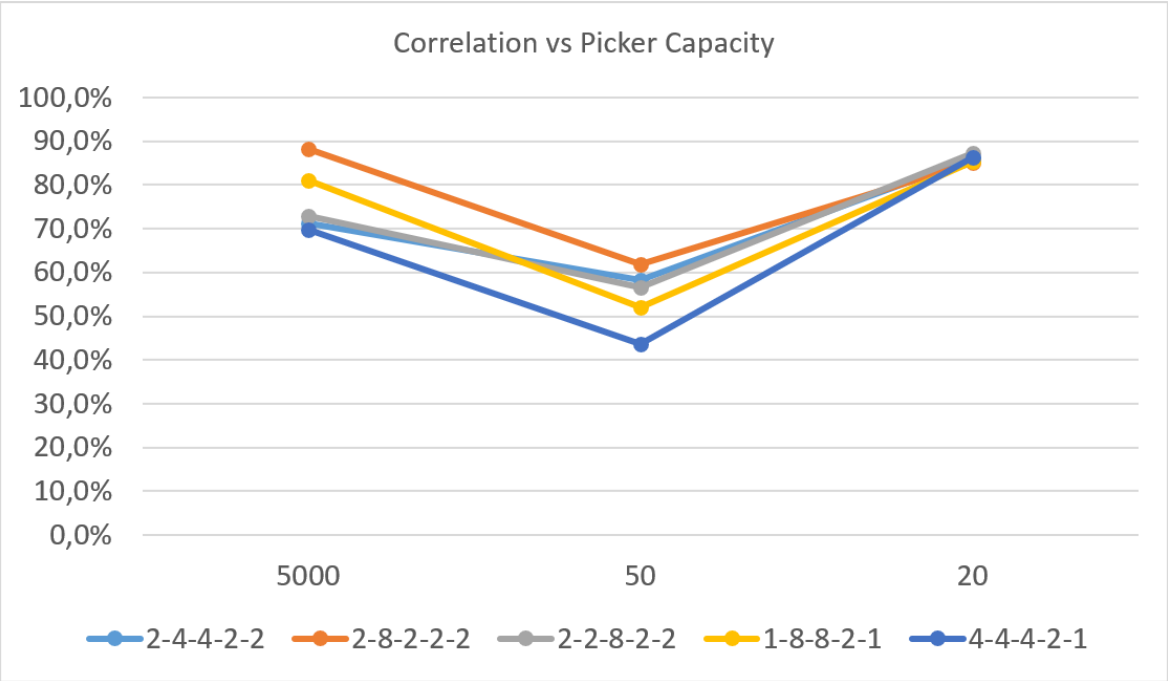

Figure 4.4. Effect of the Picker Capacity on the Correlation.

If there are single-cycle operations in a warehouse, these operations have more dominance over the distribution of the total distances traveled. For example; when we look at the 100 random allocation instances in Scenario 18, number of travels between the incoming handling area and the storage locations is 17% of total number of all visits in each of these 100 random allocation instances. This ratio is 73.4% for the travels between storage locations. On the other hand; average of total distances traveled between the incoming handling area and the storage locations is 21 distance units in the 100 random allocation instances of Scenario 18 while the average of total distances traveled between storage locations is only 6 distance units in the 100 random allocation instances of this scenario. Because of this situation, in average, the total distances traveled for the placement of the SKUs is 40.1% of the total distances traveled in Scenario 18. This ratio is 49.5% for the total distances traveled between storage locations. Also, the standard deviation of the total placement distances traveled is more than the standard deviation of the total distances traveled between the storage locations. Therefore, the dominance of placement activities is much more than their percentages by the total distances traveled.

According to the results; if number of storage locations to visit per route increases, the correlation first decreases and then starts increasing again. This is actually not surprising. If we consider that each SKU is picked and placed with only single-cycle operations, affinity becomes trivial. In that case, a greedy algorithm which considers both incoming and outgoing popularities may give a near optimal result. In the opposite case; if almost all of the storage locations in the warehouse are visited with a single picking route with no sub-routes, the picking routes become almost the same. Then, incoming popularity becomes the only significant factor. Here, how many storage locations are not visited in a route also comes into prominence. This situation explains the difference between the correlation coefficients between Scenarios 3 and 4.

It is observed that there is an almost perfect fit between the objective function values of the proposed reformulated model and the total distances traveled in a 1-1-36-2-1 type warehouse. It should be noted that it is a long warehouse sample with two long corridors. Especially when number of visits in a route is relatively high, the picker

basically enters a corridor, and at the end of this corridor, the picker enters the second one. After the last storage location is visited, the picker goes to the outgoing handling area. Since the entire warehouse is traveled, the total picking distances traveled do not change with respect to the different random allocation instances. Therefore, the only remaining factor affecting the total distances traveled becomes incoming popularity.

4.4. The Improvements Obtained By The Proposed Approach

As mentioned, the improvements of the proposed approach are determined by comparison of the total distances traveled for the allocations obtained by the proposed approach with the average and minimum of the 100 total distances traveled calculated from the 100 random allocation instances, the total distances traveled obtained by the PF/PA approach, and the total distances traveled obtained by the solutions of Bartholdi and Hackman's popularity-based greedy heuristic. The results are listed in Table 4.5.

Table 4.5. The Improvement Results of the Scenarios Studied.

Case	**vs Avg Random**	**vs Min Random**	**vs PF/PA**	**vs Greedy**
1	6.3%	4.0%	2.8%	5.2%
2	9.5%	6.3%	6.7%	9.2%
3	10.8%	7.0%	11.2%	12.3%
4	7.8%	3.4%	2.6%	11.3%
5	12.5%	9.9%	10.1%	12.1%
6	12.9%	10.3%	9.7%	13.0%
7	13.5%	11.7%	8.2%	11.6%
8	14.9%	11.4%	24.3%	24.3%
9	8.7%	6.5%	3.2%	8.0%
10	9.1%	6.7%	5.0%	8.6%
11	13.2%	8.0%	3.0%	8.6%
12	9.7%	7.4%	14.3%	16.0%
13	12.0%	10.8%	5.3%	9.7%
14	11.3%	9.2%	6.6%	9.3%
15	11.4%	10.0%	5.4%	9.0%
16	12.8%	11.0%	3.9%	8.7%
17	9.6%	8.1%	4.7%	8.8%
18	13.3%	10.7%	9.2%	13.6%
19	13.9%	11.0%	13.4%	16.0%
20	12.7%	10.0%	10.0%	12.7%
21	14.5%	12.4%	9.3%	13.8%
22	11.9%	10.0%	10.2%	13.7%
23	9.7%	8.4%	3.0%	6.2%
24	6.5%	5.0%	1.2%	2.4%
25	9.2%	7.8%	2.7%	5.0%
26	7.3%	6.2%	-0.9%	1.3%
27	7.3%	6.0%	2.2%	4.0%
28	10.5%	8.5%	0.3%	1.6%
29	13.6%	11.2%	-0.3%	0.1%

Table 4.5. The Improvement Results of the Scenarios Studied (cont.).

30	11.0%	8.8%	0.2%	1.7%
31	14.2%	11.9%	0.3%	1.5%
32	12.6%	10.3%	1.1%	2.2%
33	7.1%	5.3%	1.8%	4.2%
34	5.1%	3.8%	1.4%	2.3%
35	5.9%	4.6%	0.4%	2.5%
36	6.3%	4.0%	-0.2%	1.3%
37	7.5%	6.0%	3.4%	5.4%
38	10.6%	8.9%	0.1%	0.8%
39	11.9%	9.6%	0.0%	1.1%
40	11.2%	9.4%	0.1%	1.3%
41	12.2%	10.4%	0.0%	0.8%
42	10.1%	8.0%	0.0%	0.7%

It is observed that the proposed approach gives very applicable results in terms of the total distances traveled. In each of the 42 scenarios, the total distance traveled for the allocation obtained by the proposed approach is less than the average and minimum of the total distances traveled in the 100 random allocation instances. In average, the proposed approach obtains a 10.5% improvement compared to the average of the total distances traveled in 100 random allocation instances. Furthermore, in average, the proposed approach obtains an 8.3% improvement compared to the minimum of the total distances traveled in the 100 random allocation instances.

The proposed approach gives more applicable results for most of the examples compared to the PF/PA approach and the greedy heuristic algorithm, too. In average, the proposed approach obtains 4.8% and 8.6% improvements in the total distances traveled, compared to the PF/PA approach and the greedy heuristic algorithm, respectively. It is seen in Table 4.5 that the PF/PA approach obtains better solutions in terms of the total distances traveled in only three of the 42 scenarios, and all of these three scenarios are capacitated scenarios. Furthermore, the greedy algorithm does not obtain

any better solutions compared to the allocations obtained the proposed approach.

We see that the improvements obtained by the proposed approach in the total distances traveled are more in the incapacitated scenarios. If we consider only the total distances traveled in the incapacitated scenarios, we see that the proposed approach obtains 11.4% and 8.9% improvements compared to the average and minimum of the total distances traveled in 100 random allocation instances, respectively. Also, in average, the proposed approach obtains 8.2% and 14.1% improvements in the total distances traveled, compared to the PF/PA approach and the greedy heuristic algorithm, respectively.

There are two main reasons why the improvements that are obtained by the proposed approach are less in the capacitated scenarios, compared to the PF/PA approach and the greedy heuristic algorithm. Firstly, the abilities of the proposed approach, the PF/PA approach and the greedy heuristic algorithm to define the actual problem decrease because the information on the sub-routes are not used. Secondly, the dominance of outgoing popularity increases with decreasing picker capacity, and the total distances traveled become highly correlated with the total distances traveled for the picking operations. In these scenarios, both the proposed approach and PF/PA approach give similar results. As a result, the improvements obtained by the proposed approach in terms of the total distances traveled are less in the capacitated scenarios. Moreover, since the correlation between the total distances traveled and the objective function values of the proposed reformulated model is not 100%, both the proposed approach and the PF/PA approach may give slightly better results. Note that there is no example of this situation through the non-capacitated scenarios. This situation is seen in only three capacitated scenarios analyzed in this thesis.

The PF/PA approach and the greedy heuristic algorithm give worse results in terms of the total distances traveled in through-flow warehouses especially when the total distances traveled are dominated by the distances traveled for SKU placement activities. For instance, when the incoming and outgoing handling areas are placed in opposite locations, the greedy algorithm assigns the popular SKUs to the locations

that are close to the outgoing handling area. Additionally, PF/PA approach assigns affine SKUs to closer storage locations and popular SKUs to the storage locations close to the outgoing handling area. Since being close to the outgoing handling area means being far from the incoming handling area and the PF/PA approach considers only the picking distances traveled, the total distances traveled and the objective function values of the PF/PA model may even be negatively correlated. As a result, the results obtained by the PF/PA approach and greedy algorithm may not be better compared to even the results of the random allocation instances.

The above-mentioned situation can be seen in Scenario 19. In this scenario, the correlation coefficient between the objective function values of the reformulated proposed reformulated model and the total distances traveled is 92.7%, and a 13.9% improvement is obtained by the proposed approach compared to the average of the total distances traveled in the 100 random allocation instances. When the PF/PA algorithm is used in this scenario, the correlation coefficient between the PF/PA scores and the total picking distances of 100 random allocation instances is 71.4%. However, the correlation coefficient between the 100 PF/PA scores and the total distances traveled is -29.7% in this scenario. Furthermore, we see that there is only 0.6% improvement obtained by the PF/PA algorithm, compared to the average of the total distances traveled in the 100 random allocation instances of this scenario. Moreover, the greedy algorithm gives 2.5% worse results compared to the average of the total distances traveled in the 100 random allocation instances.

In the scenarios that there is a perfect fit between scores of the incoming popularity-related operations and the total distances traveled, the proposed approach most likely gives the optimal result since only the single-cycle operations in the SKU placement activities change the total distances traveled. In these scenarios, weight of the incoming popularity-related operations in the objective function of the proposed reformulated model is 1, and the weight parameters of the other two components of the objective function are 0. It should be noted that this makes the proposed algorithm greedy. Scenario 12 is the example of this situation. If we consider a model that minimizes only the total placement distances traveled, and SKUs are placed with only single-cycles,

a greedy algorithm can give a global optimal solution to this model. Since the total placement distances are almost fully correlated with the total distances traveled in Scenario 12, a greedy algorithm, that is only based on the total placement distances traveled, most likely gives the global optimum solution for the total distance traveled in this scenario.

5. CONCLUSION

Total distances traveled due to SKU placement and retrieval activities between the storage locations and the handling areas in through-flow warehouses are analyzed, and an approach is proposed. This approach consists of a reformulated model and a heuristic algorithm based on this reformulated model.

SLAP problem is modeled in order to optimize the allocations so that the total distance traveled due to the placement and retrieval of SKUs is minimized. Here, a reformulated model, which is derived and improved from the PF/PA model, is built to estimate the total distances traveled. The objective function of this proposed reformulated model consists of three components which are derived from the functions that formulate the scores of the incoming popularity, outgoing popularity, and affinity-related operations. Data on warehouse, SKUs, picker, stower, storage locations, and the SKU movements are generated for 42 predefined scenarios.

Firstly, 100 random storage location allocation instances are generated for each of the 42 scenarios. The total distance traveled is calculated for each random allocation instance by a routing optimization algorithm which is written for this thesis in Python and integrated with an external optimization tool, Google OR Tools. Also, scores for the three components of the objective function of the proposed reformulated model are calculated for each of the 100 random allocation instances by a objective function calculator algorithm which is also written in Python for this thesis.

Secondly, linear regression is applied on the sample data which consist of the 100 total distances traveled and scores of the three components of the objective function of the proposed reformulated model in order to obtain the weight parameters of these three components of the objective function which gives more correlated function results with the total distances traveled. Then, the finalized proposed reformulated model with the objective function with the three weight parameters is obtained.

Then, the proposed reformulated model is tried to be optimized by solving an optimization problem. Since the model does not give good results in an acceptable time period, a heuristic algorithm, namely the allocation improvement algorithm, is built to obtain good results in a shorter time. This algorithm allocates the SKUs with a greedy sub-algorithm and then improves the results via iterative reassignments. When the stoppage condition for the allocation improvement algorithm is met, the proposed allocation is obtained for the scenario analyzed.

Finally, the routing optimization algorithm is run to determine the total distances traveled. The improvements of the proposed approach are determined by comparison of the total distances traveled for the allocations obtained by the proposed approach with the average and minimum of the 100 total distances traveled calculated from the 100 random allocation instances, the total distances traveled obtained by the PF/PA approach, and the total distances traveled obtained by the solutions of Bartholdi and Hackman's popularity-based greedy heuristic. These steps are applied on each of the 42 scenarios.

The proposed approach gives appealing results in terms of the total distances traveled by the pickers and the stowers. The approach gives better solutions especially when the there is no picker capacity constraint, warehouse layout is long, and number of visits to storage locations per route is low or very high.

Further improvements may also be considered beyond this study. In order to determine the three weight parameters in the objective function of the proposed reformulated model, linear regression is applied by using only linear terms. However, it is possible that quadratic terms to be added to the regression model may improve the correlation between the total distances traveled and the objective function values of the proposed reformulated model. Also, generalized linear regression models may also be more applicable for the scenarios. In future works, these factors may be included in the algorithms and more advanced models may be obtained.

As previously stated, there are only 42 scenarios analyzed in the thesis. The main reason is that Google OR Tools obtain an optimal route in at least one second for each picklist. When there are 250 different picklists, it takes at least 250 seconds for the routing optimization algorithm to calculate the total picking distance traveled in each random allocation instance. Since there are 100 random allocation instances, it takes at least 6.9 hours to determine the 100 total distances traveled in a single scenario. This duration is calculated as 7.34 hours in scenario 16. Apart from that, other Python codes are also operated manually. In future works, if an algorithm and/or an external solver that obtains optimal routing results faster is obtained, there will be a possibility to analyze much more scenarios.

Python is a great computer language for analysts who are not software gurus since it is much easier to learn and implement compared to the other computer languages. On the other hand, there exist some other computer languages which are much faster than Python. Python is used in this thesis because the focus of the thesis is the approach itself. However, the problem can be solved in a shorter period of time by using these languages.

Multi-threading is a way that operations are done in parallel instead of waiting for the previous operation to be done to start another one. This may be implemented in the Python algorithm. Moreover, if multi-threading is used with the faster computer languages, much bigger problems may be solved by the proposed algorithm.

As pointed out, there are 42 different fictional scenarios are generated and analyzed in this thesis. The main reason here is that there is no real data could have been provided for the study. Although the total run time of the proposed algorithm is high to analyze several different scenarios, the proposed approach may be used for a single real scenario instead of 42 different scenarios. This way, total run time for the proposed algorithm may be declined epically.

The proposed approach gives rather worse results for the SLAPs when there are some constraints on the picker capacities to carry SKUs. As pointed out, the reason

is that the affinity and outgoing popularity information do not reflect the problem instances with the sub-routes well. The model may be further improved by including sub-routes information during the calculations.

The proposed approach is used in the scenarios where each SKU may be assigned to only one storage location and each storage location may be assigned to only one SKU. The approach may be improved so that it can be applied on the scenarios with more complicated allocation rules.

REFERENCES

1. Tanyaş, M. and M. Baskak, "Classification of Warehouses From Different Point of Views", *National Logistics and Supply Chain Congress*, Konya, Turkey, 2012.

2. Richards, G., *Warehouse Management: A Complete Guide to Improving Efficiency and Minimizing Costs in the Modern Warehouse*, Kogan Page, 2014.

3. Smyk, V., *Minimizing Order Picking Distance Through the Storage Allocation Policy*, Ph.D. Thesis, Arcada University of Applied Sciences, Helsinki, Finland, 2018.

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

5. Ten Hompel, M. and T. Schmidt, *Warehouse Management: Automation and Organisation of Warehouse and Order Picking Systems*, Springer, 2008.

6. Mantel, R., P. Schuur and S. Heragu, "Order Oriented Slotting: A New Assignment Strategy for Warehouses", *European Journal of Industrial Engineering*, Vol. 1, pp. 301–316, 2007.

7. Kofler, M., A. Beham, S. Wagner, M. Affenzeller and C. Reitinger, "Reassigning Storage Locations in a Warehouse to Optimize the Order Picking Process", *22nd European Modeling and Simulation Symposium*, pp. 55–76, Fes, Morocco, 2010.

8. Chuang, Y.-F., H.-T. Lee and Y.-C. Lai, "Item-Associated Cluster Assignment Model on Storage Allocation Problems", *Computers & Industrial Engineering*, Vol. 63, No. 4, pp. 1171–1177, 2012.

9. van Gils, T., K. Ramaekers, A. Caris and R. B. de Koster, "Designing Efficient Order Picking Systems by Combining Planning Problems: State-of-the-Art Classification and Review", *European Journal of Operational Research*, Vol. 267, No. 1,

pp. 1–15, 2018.

10. Kovács, A., "Optimizing the Storage Assignment in a Warehouse Served by Milkrun Logistics", *International Journal of Production Economics*, Vol. 133, No. 1, pp. 312–318, 2011.

11. Zhang, Y., Y. Yuan and K. Lu, "E-commerce Information System Data Analytics by Advanced ACO for Asymmetric Capacitated Vehicle Delivery Routing", *Information Systems and e-Business Management*, Vol. 18, No. 4, pp. 911–929, 2020.

12. Özkaraca, O., "A Review on Usage of Optimization Methods in Geothermal Power Generation", *Mugla Journal of Science and Technology*, Vol. 131, No. 1, pp. 130–136, 2018.

13. Kofler, M., *Optimising the Storage Location Assignment Problem Under Dynamic Conditions*, Ph.D. Thesis, Johannes Kepler University, Linz, Austria, 2015.

14. Conforti, M., G. Cornuéjols and G. Zambelli, *Integer Programming*, Springer, 2014.

15. Aras, N. and B. Görgülü, "An Adaptive Large Neighborhood Search Heuristic for Jointly Solving Storage Location Assignment and Picker Routing Problem", *Operations Research Proceedings 2018*, Brussels, Belgium, 2018.

16. Evinsel, C., *Warehouse Design and Layout*, Master's Thesis, Istanbul University, Istanbul, Turkey, 2010.

17. Battista, C., A. Fumi, F. Giordano and M. Schiraldi, "Storage Location Assignment Problem: Implementation in a Warehouse Design Optimization Tool", *Breaking Down the Barriers Between Research and Industry Proceedings*, Padua, Italy, 2011.

18. Zhang, Y., "Multi-Objective Evolutionary Algorithms of Correlated Storage Assignment Strategy", *International Conference on Modeling and Applied Simulation Proceedings*, pp. 16–24, Larnaca, Cyprus, 2016.

19. Yang, M.-H., "An Efficient Algorithm to Allocate Shelf Space", *European Journal of Operational Research*, Vol. 131, No. 1, pp. 107–118, 2001.

20. Ozyoruk, B. and S. Ak, "A Model to Arrange Warehouse Layout Effectively: Application in an Electronic Firm", *Journal of TUBAV Science*, Vol. 5, No. 1, pp. 21–29, 2012.

21. Ayhan, M. B., S. Bulkan, M. Bilsel and A. Gülcü, "Storage Location Space Optimization", *Operations Research/Industrial Engineering National Congress Proceedings*, Izmir, Turkey, 2007.

22. Wang, M. and R.-Q. Zhang, "A Dynamic Programming Approach for Storage Location Assignment Planning Problem", *11th CIRP Conference on Industrial Product-Service Systems Proceedings*, Vol. 83, pp. 513–516, Zuhai and Hong Kong, China, 2019.

23. Çolak, M., G. Aydın Keskin, G. Günel and D. Akkaya, "A Model for Efficient Layout of a Raw Material Warehouse in a Chemical Company", *Beykent University Journal of Science and Engineering*, Vol. 9, No. 2, pp. 55–76, 2016.

24. Liu, C.-M., "Optimal Storage Layout and Order Picking for Warehousing", *International Journal of Operations Research*, Vol. 1, No. 1, pp. 37–46, 2004.

25. Mirzaei, M., N. Zaerpour and R. de Koster, "The Impact of Integrated Cluster-Based Storage Allocation on Parts-to-Picker Warehouse Performance", *Transportation Research Part E: Logistics and Transportation Review*, Vol. 146, 2021.

26. *OR Tools Python Integration*, `https://developers.google.com/optimization/routing/vrp`, Accessed in January 2022.

27. *About OR-Tools*, `https://developers.google.com/optimization/introduction/overview`, Accessed in January 2022.

28. *Get Started Guides*, `https://developers.google.com/optimization/introduction/get_started`, Accessed in January 2022.

29. *Routing Options*, `https://developers.google.com/optimization/routing/routing_options`, Accessed in January 2022.

30. Martí, R., P. M. Pardalos and M. G. C. Resende, *Handbook of Heuristics*, Springer, 2018.

APPENDIX A: VERIFICATION

Since there are lots of code lines written for the proposed algorithm, whether the algorithm operates correctly needs to be analyzed. To do so, results are analyzed on a smaller model which is defined for testing purposes. Following conditions are checked to figure out whether the algorithm operates correctly:

Affinity matrix:

- Affinity matrix is symmetric.
- Incoming popularity of an SKU is equal to total number of its entries to the warehouse.
- Outgoing popularity of an SKU is equal to number of occurrences in the picklists.
- Outgoing popularity of the phantom SKU for the outgoing handling area is equal to total number of the picklists.
- Score of the incoming popularity-related operations for an allocation instance is equal to the half of the total distance traveled for stowing activities. The reason here is that the stower is assumed to be operating with single-cycles. Since the path between the storage location and the handling area is used twice, the distance traveled to stow an SKU becomes twice of the distance between them. Of course, there will be a perfect fit between score of the incoming popularity-related operations for an allocation instance and the total distance traveled for the placement of the incoming SKUs.

Distance matrix:

- Affinity matrix is symmetric.

Random allocation instances:

- Each SKU is assigned to only one storage location.
- Each storage location assigned to only one SKU.
- There is neither an unassigned SKU nor an unassigned storage location when the assignments are complete.
- Phantom SKUs are assigned to their corresponding phantom storage locations.

Allocation:

- When weight of the incoming popularity-related operations equals to 1 and the others are set to 0, there is a perfect fit between the total distances traveled for the placement of the incoming SKUs and objective function values of the proposed reformulated model. Also, the allocation algorithm becomes purely greedy. It sorts the SKUs in terms of the incoming popularities in descending order and assigns them to the storage locations from the closest to the farthest.
- When all the three weight parameters of the objective function value of the proposed reformulated model are multiplied by a positive number at the same time, the allocation algorithm still obtains the same solution.

Routing:

- Distance traveled in each of the randomly selected routes fully reflects the distances defined in the distance matrix.
- When the picker capacity is full, the next location to visit of the picker is the outgoing handling area.

The routing optimization algorithm is the one which uses most of the other codes as input. Moreover, this algorithm is more complicated and it is integrated with Google OR Tools. Since its verification can not be done by a simple manual check, a sample case is analyzed using the test mode.

In Table A.1, the storage locations and indices of the them are defined. The storage locations are defined by these indices in the distance matrices for the table to fit the page.

Table A.1. Sample Storage Location Indices.

Storage Location	**Indice**
R000	1
R1-01-01-1-1	2
R1-01-01-2-1	3
R1-01-02-1-1	4
R1-01-02-2-1	5
R1-01-03-1-1	6
R1-01-03-2-1	7
R1-02-01-1-1	8
R1-02-01-2-1	9
R1-02-02-1-1	10
R1-02-02-2-1	11
R1-02-03-1-1	12
R1-02-03-2-1	13
R999	14

In this part, three of the proposed routes that are obtained by the routing optimization algorithm are randomly selected. The distance traveled for each of these routes is manually calculated, and compared with the distance traveled which is calculated by the routing optimization algorithm. Also, whether the capacity constraint of the picker, whose upper bound is assumed to be 20 SKUs for the test case, is not violated is checked.

A.1. Route 1: A Picklist

There are six SKUs with a total amount of 46 in the first picklist. The SKUs in the picklist with their quantities and assigned storage locations are illustrated in Table A.2.

Table A.2. Sample Order Information of the First Scenario.

SKU	**QTY**	**Location**
S002	3	R1-02-03-2-1
S004	8	R1-01-03-1-1
S001	3	R1-02-02-2-1
S010	1	R1-02-03-1-1
S003	4	R1-01-02-2-1
S011	5	R1-01-01-2-1

The route obtained for this scenario is defined in Table A.3. As can be seen from the table, the picker capacity constraint is not violated.

Table A.3. Sample Routing Information of the First Scenario.

Route	**Qty Before Visit**	**Qty Picked**	**Qty After Visit**
R000	0	0	0
R1-01-03-1-1	0	8	8
R000	8	0	0
R1-01-01-2-1	0	5	5
R1-01-02-2-1	5	4	9
R1-02-03-1-1	9	1	10
R1-02-02-2-1	10	3	13
R1-02-03-2-1	13	3	16
R000	16	0	0

The paths that are used during a route for picking SKUs can be seen in Table A.4 with their corresponding sub-route indices attached. Also, it may be seen that the distance matrix is symmetric.

Table A.4. Sample Distance Matrix and the Visited Storage Locations in the First Scenario.

Indice	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
1	0.00	3.75	6.75^{2}	5.25	8.25	6.75^{1}	9.75	9.75	12.75	11.25	14.25	12.75	15.75	19.50
2	3.75	0.00	4.50	1.50	6.00	3.00	7.50	7.50	10.50	9.00	12.00	10.50	13.50	15.75
3	6.75	4.50	0.00	6.00	1.50^{2}	7.50	3.00	3.00	7.50	4.50	9.00	6.00	10.50	12.75
4	5.25	1.50	6.00	0.00	7.50	1.50	9.00	9.00	12.00	10.50	13.50	12.00	15.00	14.25
5	8.25	6.00	1.50	7.50	0.00	9.00	1.50	4.50	9.00	3.00	10.50	4.50^{2}	12.00	11.25
6	6.75^{1}	3.00	7.50	1.50	9.00	0.00	10.50	10.50	13.50	12.00	15.00	13.50	16.50	12.75
7	9.75	7.50	3.00	9.00	1.50	10.50	0.00	6.00	10.50	4.50	12.00	3.00	13.50	9.75
8	9.75	7.50	3.00	9.00	4.50	10.50	6.00	0.00	4.50	1.50	6.00	3.00	7.50	9.75
9	12.75	10.50	7.50	12.00	9.00	13.50	10.50	4.50	0.00	6.00	1.50	7.50	3.00	6.75
10	11.25	9.00	4.50	10.50	3.00	12.00	4.50	1.50	6.00	0.00	7.50	1.50	9.00	8.25
11	14.25	12.00	9.00	13.50	10.50	15.00	12.00	6.00	1.50	7.50	0.00	9.00	1.50^{2}	5.25
12	12.75	10.50	6.00	12.00	4.50	13.50	3.00	3.00	7.50	1.50	9.00^{2}	0.00	10.50	6.75
13	15.75^{2}	13.50	10.50	15.00	12.00	16.50	13.50	7.50	3.00	9.00	1.50	10.50	0.00	3.75
14	19.50	15.75	12.75	14.25	11.25	12.75	9.75	9.75	6.75	8.25	5.25	6.75	3.75	0.00

According to the manual calculations, the distance traveled for the all SKUs in the picklist to be retrieved is 52.5 distance units. The routing optimization algorithm also found this result.

A.2. Route 2: A Picklist

There are 12 SKUs with a total amount of 46 in the second picklist. The SKUs in the picklist with their quantities and assigned storage locations are illustrated in Table A.5.

Table A.5. Sample Order Information of the Second Scenario.

SKU	**QTY**	**Location**
S012	2	R1-01-01-1-1
S008	4	R1-01-02-1-1
S004	6	R1-01-03-1-1
S003	1	R1-01-02-2-1
S010	2	R1-02-03-1-1
S005	5	R1-02-01-2-1
S007	3	R1-02-02-1-1
S009	3	R1-01-03-2-1
S006	3	R1-02-01-1-1
S001	8	R1-02-02-2-1
S011	5	R1-01-01-2-1
S002	4	R1-02-03-2-1

The route obtained for this scenario is defined in Table A.6. As can be seen from the table, the picker capacity constraint is not violated.

Table A.6. Sample Routing Information of the Second Scenario.

Route	**Qty Before Visit**	**Qty Picked**	**Qty After Visit**
R000	0	0	0
R1-02-01-2-1	0	5	5
R1-02-03-2-1	5	4	9
R1-02-02-2-1	9	8	17
R000	17	0	0
R1-01-01-1-1	0	2	2
R1-01-02-1-1	2	4	6
R1-01-03-1-1	6	6	12
R000	12	0	0
R1-01-01-2-1	0	5	5
R1-02-01-1-1	5	3	8
R1-02-02-1-1	8	3	11
R1-02-03-1-1	11	2	13
R1-01-03-2-1	13	3	16
R1-01-02-2-1	16	1	17
R000	17	0	0

The paths that are used during a route for picking SKUs can be seen in Table A.7 with their corresponding sub-route indices attached. Also, it may be seen that the distance matrix is symmetric.

Table A.7. Sample Distance Matrix and the Visited Storage Locations in the Second Scenario.

Indice	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
1	0.00	3.75^{2}	6.75^{3}	5.25	8.25	6.75	9.75	9.75	12.75^{1}	11.25	14.25	12.75	15.75	19.50
2	3.75	0.00	4.50	1.50^{2}	6.00	3.00	7.50	7.50	10.50	9.00	12.00	10.50	13.50	15.75
3	6.75	4.50	0.00	6.00	1.50	7.50	3.00	3.00^{3}	7.50	4.50	9.00	6.00	10.50	12.75
4	5.25	1.50	6.00	0.00	7.50	1.50^{2}	9.00	9.00	12.00	10.50	13.50	12.00	15.00	14.25
5	8.25^{3}	6.00	1.50	7.50	0.00	9.00	1.50	4.50	9.00	3.00	10.50	4.50	12.00	11.25
6	6.75^{2}	3.00	7.50	1.50	9.00	0.00	10.50	10.50	13.50	12.00	15.00	13.50	16.50	12.75
7	9.75	7.50	3.00	9.00	1.50^{3}	10.50	0.00	6.00	10.50	4.50	12.00	3.00	13.50	9.75
8	9.75	7.50	3.00	9.00	4.50	10.50	6.00	0.00	4.50	1.50^{3}	6.00	3.00	7.50	9.75
9	12.75	10.50	7.50	12.00	9.00	13.50	10.50	4.50	0.00	6.00	1.50	7.50	3.00^{2}	6.75
10	11.25	9.00	4.50	10.50	3.00	12.00	4.50	1.50	6.00	0.00	7.50	1.50^{3}	9.00	8.25
11	14.5^{1}	12.00	9.00	13.50	10.50	15.00	12.00	6.00	1.50	7.50	0.00	9.00	1.50	5.25
12	12.75	10.50	6.00	12.00	4.50	13.50	3.0^{3}	3.00	7.50	1.50	9.00	0.00	10.50	6.75
13	15.75	13.50	10.50	15.00	12.00	16.50	13.50	7.50	3.00	9.00	1.5^{1}	10.50	0.00	3.75
14	19.50	15.75	12.75	14.25	11.25	12.75	9.75	9.75	6.75	8.25	5.25	6.75	3.75	0.00

According to the manual calculations, the distance traveled to retrieve the all SKUs in the picklist is 630 distance units. The result is the same in the routing optimization algorithm.

A.3. Route 3: Placement of an SKU

As pointed out in Chapter 2, SKU placements are done with single-cycles. Thus, the Python code considers each row in the SKU list for placement as separated lists with a single row.

In the placements of SKUs, each routing distance is twice the distance between the assigned storage location and the incoming handling area. As stated before, this situation is normal in single-cycle processes. The routing optimization algorithm also obtains the same result. Also, these results are added to the cumulative total distance traveled correctly.