AN INTEGRATED STOCK ASSIGNMENT MODEL FOR A WAREHOUSE FAST PICKING AREA

A Thesis

presented to

the Faculty of the Graduate School

at the University of Missouri-Columbia

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

---

by

SAMUEL CUAUHTLI-OLLIN TREVIÑO MARTINEZ

Dr. James Noble, Thesis Supervisor

DECEMBER 2008

The undersigned, appointed by the dean of the Graduate School, have examined the thesis entitled

AN INTEGRATED STOCK ASSIGNMENT MODEL FOR
A WAREHOUSE FAST PICKING AREA

presented by Samuel Cuauhtli-Ollin Treviño Martinez,

a candidate for the degree of master of science,

and hereby certify that, in their opinion, it is worthy of acceptance.

Professor James Noble

Professor Alec Chang

Professor Lori Franz

…….Thanks to my Mother, Martha Martinez, and my Father, Jaime Trevino, for always supporting me in my decisions and giving me great advices in my life. To my uncle, Miguel Angel Martinez, for encouraging me throughout my graduate studies and his great advice in applying for the Fulbright Program. And to my girlfriend, Stacy King, for her unconditional support and care during this two years of my degree. Finally, I dedicate this research to my brother, Leon Tonatihu, and sister, Alicia Martinez, to motivate them in finding excitement in their future studies.

ACKNOWLEDGEMENTS

First, I would like to thank my family for their support and inspiring me in going into graduate school. I am also thankful to the Fulbright Program and the COMEXUS commission in Mexico for their generous scholarship grant that allowed me to pursue a Master's degree program in the United States.

I want to express my appreciation to my academic supporters. I thank my advisor, Dr. James Noble, who awarded me the opportunity to join the CELDi (Center for Engineering Logistics and Distribution) – Hallmark project where I found the motivation to conduct this research. I also thank my co-advisors, Dr. Alec Chang and Dr. Lori Franz, for encouraging me; and professor Dr. Mustafa Sir for his valuable advices in operations research.

I am thankful for the opportunities given by Hallmark Inc. and Scholastic Inc. to collaborate with them in their distribution center, which help me broaden my understanding of a fast picking area and capture the problem structure that became the foundation of this research.

I also give special thanks to my friends and officemates: Gaohao Luo, who supported me with advices in LINGO software; Almas Ospanov, for his comments and support; Phichet Wutthisirisart for his tremendous help in the CELDi-Hallmark project and his thoughts in order sequencing algorithms; and Wilfred Fonseca for his help in programming and insights on mathematical formulations.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

An Integrated Stock Assignment Model for a Warehouse Fast Picking Area

Samuel Cuauhtli-Ollin Treviño Martinez

Dr. James Noble, Thesis Supervisor

ABSTRACT

Warehousing is a significant component of the supply chain activities. Modern distribution centers configure their warehouses in fast picking areas to fulfill orders more rapidly at a lower cost while achieving customer satisfaction. Order picking and restocking of the fast picking area have been identified as the most labor-intensive and costly activities of any distribution center. In a picker-to-item process environment, traveling is an important factor than can account up to 50% of the total labor time dedicated to order picking and restocking. Previous research has mostly focused in reducing costs from a myopic perspective, either analyzing the flows from order picking or restocking. This research integrates order picking and restocking to generate a stock keeping unit (SKU) layout that minimizes the overall walking distance traveled within an S-shaped routing policy. Given a set of order routes and SKU restocking frequencies, the assignment of SKUs to a location is formulated as a Binary Mixed Integer Linear Programming (BMILP) model which is able to solve small scale problems to optimality. In the cases where the BIMLP model does not find the optimal solution, the best feasible solution falls within 20% of the lower bound. For larger scale problems, a heuristic is presented obtaining solutions in little computation time. The BMILP and the heuristic are compared to other scientific and popular methods in practice, and show that additional savings in labor can be obtained.

CHAPTER 1

INTRODUCTION

As today's market environment forces companies to be competitive, the supply chain has been under increased study looking for improvements in costs, quality, delivery, flexibility and customer satisfaction. As illustrated in Figure 1.1, distribution and logistics is a significant subsystem of the overall supply chain activities. The supply chain costs are equivalent to 10% of the U.S. gross domestic product (GDP) demonstrating the huge impact it posses in the industry [1].

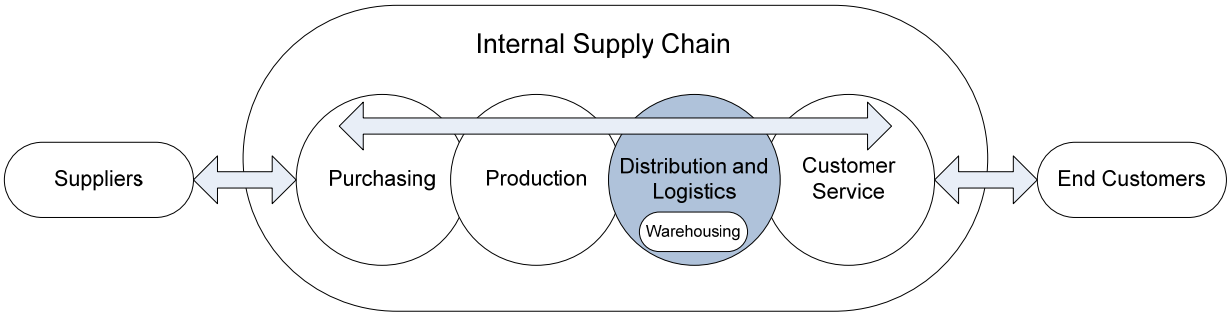


Figure 1.1 An illustration of a company's supply chain; the arrows represent the relationship flow.

Warehousing is a critical component for distribution and logistics. Warehouses are mostly used to: 1) protect the product from the environment; 2) consolidate products to reduce transportation costs, 3) respond quickly to changes in demand; and 4) provide value-added processing such as light assembly used in customized orders. Requiring substantial labor, capital and information systems, warehousing systems are expensive, accounting for 20-25% of the total

distribution and logistics operational expenses (Frazelle, 2002). According to Goldratt (2004), the three bottom line performance measurements any company should observe are throughput, inventory, and operating expense. Thus, improvements in the planning and control of warehousing systems are essential for the success of any enterprise's supply chain. Warehouse systems planning includes: operating technology planning, equipment selection, layout design, space allocation, and strategies for assigning products to storage locations. Control of a warehouse system involves sequencing, scheduling, re-profiles, and routing strategies. In this paper, the layout design and product assignment strategies in warehouse planning are given more attention per the systems thinking assumption that better planning would lead to a better control.

1.1 Warehouse Structure

The main functional areas in a warehouse are: receiving, stock storage, replenishment, order-picking, quality control, sortation and consolidation, and loading, see Figure 1.3.

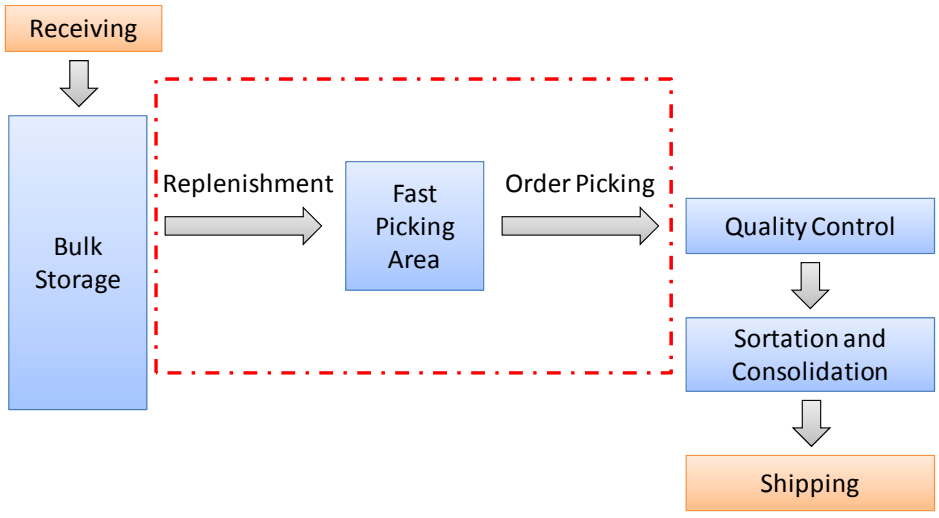


Figure 1.2 Main work flow structure of a warehouse.

The *receiving* area involves unloading products from the supplier's transport carrier, properly identifying and inspecting the quality of the product, updating the inventory level and preparing the product to be transferred to a storage location in the warehouse.

The *replenishment* function transfers and puts away the product to a storage location. Depending on the warehouse configuration, replenishment is usually broken into two areas: reserve replenishment, and fast picking area replenishment. The first puts away any receiving items into the bulk warehouse area, and the second replenishes the fast picking area from the bulk warehouse.

Order-picking is the main core of the warehouse operation. Its task is to fulfill the set of customer orders by reaching the right item location, obtaining the right amount, and putting them into the order box and properly labeling it.

Quality control assures that the orders picked by the order-pickers are in compliance with the customer orders. If orders are in compliance, they continue to be sorted and consolidated; otherwise, they are sent back to the order pickers.

Sorting and Consolidation involves transferring the finished order to the outgoing door for shipment. Prior to shipment, the picked orders are grouped and stacked to be palletized. The pallets are then labeled accordingly, and staged close to the outgoing door waiting to be *load* into the shipping carrier.

1.2 Order Picking

Several order picking systems exist nowadays, creating different warehousing environments to increase output productivity. Three main systems are distinguished by their order picking approach: *picker-to-item, item-to-picker,* and *automated picking.*

In a *picker-to-item* system, order pickers need to travel along the rack shelves to retrieve the items in their order lists. Three main methods of retrieval are observed, 1) one in which the orders are picked one at a time; 2) items are sorted during the picking route; and 3) items are picked in a single container and later at the end of the route sorted into their respective orders. The first method is widely used when orders are small or when the picking area is divided into zones. A picking area divided into zones requires pickers to only pick items located in their zones and pass the order to the next zone. Usually, the orders are transferred from zone to zone by means of a powered conveyor. The second method requires batching. Batch picking is very practical in warehouses since it allows several pickers to work on different orders, maximizing the throughput performance of the order pickers. Batching is also necessary when orders are very large, making it unfeasible to be picked by a single picker. The third method is seen more when the items are similar among all the orders, making it easier for pickers to move from location to location. This is similar to a consolidation problem, where similar products are picked together and then distributed to different customers.

One of the technologies widely used in *picker-to-item* environments is the *pick-to-light* and *pick-to-route* technology. The pick-to-light technology is mostly used in picking zones, where the pickers scans the picking list and the item location lights up displaying the quantity to be retrieved. Pick-to-route technology is mostly used in batch picking, where the set of batched

orders are loaded into a portable system, then, by display or audio means the item location to visit is given along with the quantity to retrieve.

In an *item-to-picker* environment, the picker enters the set of orders required and waits for the item to be retrieved and reach to him. This picking environment is seen in cranes, AS/RS (Automated Storage and Retrieval Systems), Vertical Lift Modules (VLM), or carrousel systems, in which the items are picked as a unit load and received at the input/output point.

New trends in technology development have led to many *automated picking* machines. An example is the A-frame, consisting of dispenser channels, where each channel holds a supply of at most one *stock keeping unit* (SKU). A conveyor passes underneath the row of channels containing the order, and if the SKU is required in the order, the right amount of the SKU is dispensed into the order.

1.3 Replenishment

Order picking performance is highly impacted by the replenishment function. The order picking areas need to be constantly kept with right inventory levels to meet customer demand. Inefficient or inaccurate replenishment can lead to on-hold orders, meaning that an order could not be fulfilled due to product unavailability at the picking location. On-hold orders are one of the most unwanted incidents at any distribution center. Not only will they slow down order picking and replenishment productivity, but they could also generate extra-costs associated with backordering. Backordering extra costs involves: overtime labor for pickers and restockers, shipping expedites, and customer un-satisfaction. Due to this, the planning of the replenishment

system is crucial for the accurate flow of items from the warehouse to the order picking locations.

In typical distribution centers, the replenishment flow occurs either in full-pallet or single cases. *Full-pallet replenishment* is similar to cross-docking; the pallet is staged in the back of the warehouse and then moved to a more convenient location in the order picking area. This strategy minimizes the labor involved in extracting the item from the back of the warehouse, however, such labor savings may be minimized at front when the pallet needs to be broken into single cases to replenish the picking locations, which is the case for order picking gravity-racks. In a DC where there are thousands of SKUs, full-pallet pulls is not a feasible strategy due to space constraints. In addition, an abuse of full pallets can overload the workload of restockers in the picking area. As a result, the decision of which items to full-pull becomes critical when applying full-pallet replenishment.

Single-case replenishment, contrary to full-pallet replenishment, deals with more labor in the back of the warehouse increasing the time to extract the necessary number of cases to replenish the picking area, but reduces the labor of restockers in the front picking area. When extracting the items to be replenished in the front picking area, the case picker consolidates the single case replenishment orders to single pallets. Afterwards, the single pallets are staged for delivery to the picking area. The internal delivery flow to the picking are can be automated or manual. In an automated replenishment flow system, an operator places the cases from the pallets onto a conveyor belt which diverts the cases to their respective picking areas. Later, restockers at the picking area wait for the cases and put them away in their respective locations. Manual flow involves transferring single case pallets to their picking areas by means of a fork-lift or pallet-jack.

1.4 Fast Picking Area

Due to an increase of order customization with fluctuating demands, modern distribution centers configure their warehouses with fast picking areas to fulfill orders more rapidly at a lower cost and to increase order responsiveness to customer's changes in demand. Under this strategy, the warehouse is dived into a reserve area and a forward area, also called fast picking area. The fast picking area may store only single case quantities of some or all SKUs, thus, increasing the SKU pick density significantly and reducing the amount of time associated with order picking. The reserve area is then used to store bulk pallets coming from vendors and to replenish the fast picking area. In some situations, items that are not assigned to the fast picking area are picked from the reserve area.

The fast picking area may not be a separate area from the reserve all the time. In some cases, the fast picking area and the reserve may share the space, i.e., the fast picking area may be in the bottom of the racks and the reserve area on top.

Figure 1.4 illustrates the SKU flow in a forward-reserve warehousing configuration strategy, which is actuated by the replenishment and order picking functions. From the replenishment side, incoming items are received and put away into pallets in the reserve area storage locations. As new order sets are received from customers, items are pulled from the reserve area to replenish the fast pick area. Depending on the replenishment system used, the replenishment can happen ahead of the picking schedule or during the order picking process. On the order picking side, customer orders are fulfilled from the fast picking area and sent to the shipping area. The orders that contain items that are not found in the fast pick area are then picked from the reserve area.

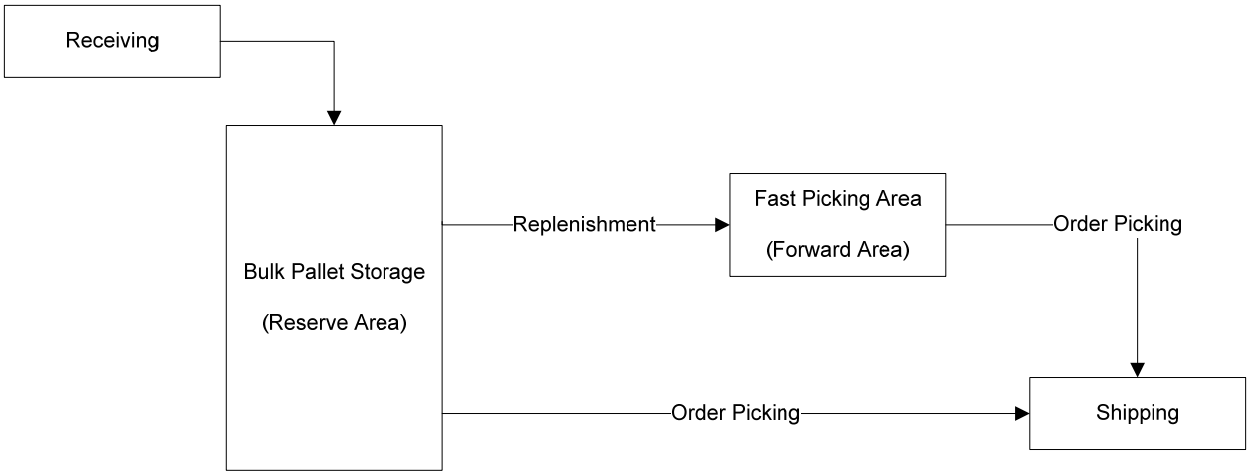


Figure 1.3 A forward-reserve warehouse configuration.

There are important considerations to be taken into account when adopting a fast picking area strategy. From a cost perspective, while it is true that a fast picking area reduces labor time dedicated to order picking, it is also a fact that it requires more frequent replenishments from the reserve area, thus, increasing the replenishment labor. In addition to this more space and increased material handling equipment investment is required. These aspects create non-trivial tradeoffs between savings in the order picking process versus impact on replenishment labor. These tradeoffs have an inverse relationship between the size of the fast picking area and the number of replenishments. As the size of the fast picking area increases, the order picking costs increase and the replenishment costs decrease, and vice-versa. Deciding which items to place in the fast picking area and finding the optimal space allocation for each of them such that it minimizes the order picking, replenishment and equipment costs is known as the Forward/Reserve Problem (FRP), first introduced by Hackman and Rosenblatt (1990).

1.5 Focus of Research

Order picking and replenishment have been identified as the most labor intensive and costly activities of any fast picking in a distribution center. In a typical warehouse environment, 70% of the total operating cost is attributed to order picking and replenishment functions. Thus, improvements in the order picking and replenishment function are of great interests. Within a fast picking area, traveling is the most time consuming activity, shown in Figure 1.5. Therefore, minimizing the travel distance is expected to significantly reduce the labor costs of the fast picking area and benefit the overall performance of the warehouse by fulfilling orders more rapidly and increasing the productivity of restockers.

Due to the high relationship that exists between order picking and replenishment, this research integrates them to generate a SKU layout that minimizes the overall walking distance traveled. The fast picking area warehouse configuration chosen for this research consists of a picker-to-item picking environment with batch picking and an automated single-case replenishment system.

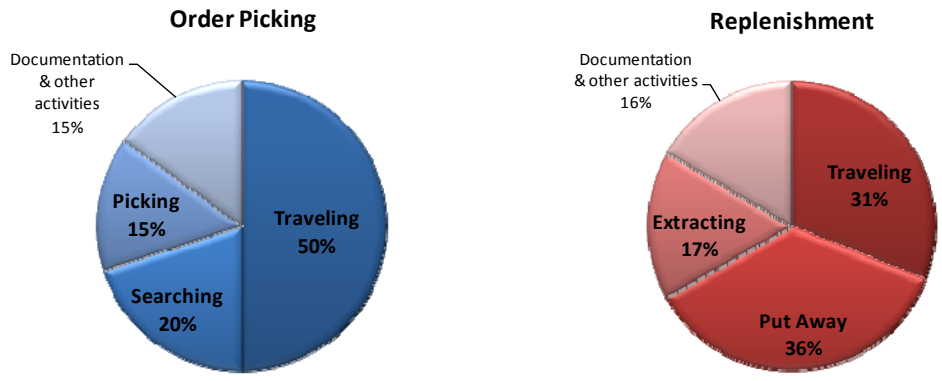


Figure 1.4 Labor costs in a fast picking area.

In the next chapter, a literature review is presented that describes previous work done on warehousing layout design, order picking slotting strategies, and replenishment systems with the objective to reduce labor costs in warehousing.

Chapter 3 introduces the problem structure of the fast picking area to be analyzed. Then an integrated mixed integer linear programming model to optimize the total traveling distance by order pickers and restockers is presented. Finally, a heuristic is proposed that solves large scale problems close to reality in a reasonable amount of time and with outstanding quality.

Chapter 4 presents a sensitivity analysis to compare the solution quality and solution time obtained by both the mathematical model and the heuristic. Results are compared to other approaches formed in the literature and it is shown that significant savings in labor can be obtained with the proposed model.

Chapter 5 summarizes and concludes this research followed by proposed future research directions for the optimization of traveling distances within a fast picking area.

CHAPTER 2

LITERATURE REVIEW

As discussed in section 1.5, the objective of this research is to reduce the labor costs of a DC's fast picking area by tackling the traveling distances incurred in order picking and replenishment functions. In a DC, the traveling distances by order pickers and restockers is affected by four main warehousing factors: 1) material handling technology; 2) warehouse layout; 3) routing; and 4) SKU slotting.

Material handling technology is perhaps the basis under which warehousing planning has been done lately. In many cases it has been a strong constraint in a DCs layout configuration planning. This requires the layout to work around the material handling system used, (i.e. if the chosen replenishment flow involves conveyors, the order picking zones need to be close to the conveyor diverts and the distance between aisles would need to be longer to allow the conveyor space requirements).

The warehouse layout involves the decision of: a) number of racks and aisles –including length and width of aisles; b) orientation of aisles and picking racks; and c) location of input/output (I/O) points, which are the starting traveling points for order pickers and restockers, and incoming/outgoing doors dedicated for shipping.

The travel distance by order pickers and restockers is significantly affected by the configuration chosen, i.e., if the aisles are too long, there would be potentially more walking involved within

the aisles, if too short, the traveling distance between aisles increases (see Roodbergen et al., 2006, and Alagoz et al., 2008)

Routing concerns the traveling direction of order pickers from location to location to retrieve the items needed in the order. The routing policy becomes crucial in reducing the amount of idle walk involved during the order picking process. Four routing methods are described in De Koster et al. (2007): *S-shape*, *midpoint*, *largest gap*, and *combined*.

SKU slotting planning is also known as the SKU assignment problem. Due to the fact that order pickers have to reach the location of the item to be picked and restockers have to put away the boxes of items in their respective slot, the location assigned to each item will determine the amount of distance to be at least traveled in the picking and restocking process. The slotting decision concentrates on the flow of items, mainly dictated by demand and carton size, in order to assign a location or zone to every item that is picked in the fast picking area.

Unlike the rest of the warehousing factors previously discussed, SKU slotting offers great advantages in the design of a warehouse since it works at the SKU level –the core of order picking and replenishment- and, thus, allows dynamic adjustments in the fast picking floor as SKU velocity changes during the year without needing to reconfigure the warehouse structure or change the material handling equipment. Moreover, a good slotting strategy will benefit any routing policy.

2.1 Slotting Policies in a Picking Area

There are numerous ways one could assign products to storage locations in a fast picking area. Hausman et al. (1976) present three storage assignment policies which are nowadays very

popular in warehousing management: *random storage*, *class-based storage*, and *dedicated storage*. The random storage assignment policy allows any incoming product to be stored anywhere in the picking area. A class-based assignment policy reserves a space in the picking area which can be shared by a set of defined product groups. The dedicated storage assignment policy gives to every product a unique area which cannot be occupied by any other product. Issues related to these three assignment policies are described next.

2.1.1 Random Assignment

The concept of randomized storage is to assign to every incoming product a location, randomly selected from all the available locations with equal probability randomly, in the fast picking area (see Petersen, 1997). The random storage policy would only work in a computerized warehousing environment in order to keep track of the locations assigned to each product. Otherwise, the search-time of the items during the retrieval process would be seriously affected.

A variation of the random storage assignment policy is the *closest open location storage*, discussed by De Koster et al (1997), in which the first empty location encountered by an employee during the put away process becomes a candidate location for the incoming product.

The greatest advantage of random storage policy is its high space utilization, since any available space in the picking area becomes a candidate to store any incoming product, thus, minimizing the space requirements. Also, in terms of productivity, the put away labor is reduced, which can be beneficial when there is a constant flow of incoming products that need to be put away as soon as possible. Nevertheless, the amount of traveling involved in the retrieval of products by a manual picker is very likely to increase. For such reasons, randomized storage is more practical in *automated picking* environments as: AS/RS, VLM's, or carousels.

2.1.2 Class Based Assignment

The class-based storage policy is a combination of random and dedicated storage policies. The idea behind it is to divide the inventory items into classes. Each class would have an assigned area, where any space available within it is randomly occupied by the products belonging to that class. Randomized and class-based storage are also known as *shared storage policies*, for these allow different products to successively occupy the same location.

The generation of classes can occur in many ways. From an inventory management perspective, classes can be determined by the products life, i.e. perishable items could be grouped together versus non-perishable, or by their incoming sequence, i.e. first-in first-out (FIFO) and last-in first-out (LIFO). The most common used class-based strategy follows Pareto's popularity method; based on the assumption that 20% of the products contributes to 80% of the sales or pick volume. The arrangement of classes based on the products popularity would lead to the widely used ABC layout policy (see Le-Duc and De Koster, 2005). In it, the most popular class or fast movers are the A-items, and would take over the most convenient locations in the warehouse; typically close to the I/O points or outgoing shipping door. The next fastest moving class is the B-items, and take the next most convenient zone locations after A-items. The next class would be C-movers, and so on. Although the class-based storage policy assigns the fast-moving items closer to the I/O points, the space requirements increase with the number of classes. For these reason, no more than three classes are typically used.

2.1.3 Dedicated Assignment

Due to the nature of fast picking areas where every item needs to be assigned a specific location, as would be in the case of single-item gravity racks, dedicated storage is the most used policy in

DC's. Five dedicated assignment methods are discussed next: *Cube-per-Order Index*, *Correlated Assignment*, *Quadratic Assignment Problem*, *Order-Orienting Slotting*, and the *Forward-Reserve Problem.*

2.1.3.1 Cube per Order Index

One of the oldest and most popular assignment methods for the dedicated storage systems is the Cube-per-Order Index (CPO), introduced by Heskett (1963). In the literature, two basic elements in determining the stock location are described: stock size and product popularity. On one hand, the stock size defines the space required to be allocated to each product; on the other, product demand determines the turnover of the location, and the average number of times the product appears on orders over a period of time. Hence, the COI can be defined as the ratio of the product's required space to the number of trips required to fulfill the products order demand per period – the basis of Heskett's (1964) algorithm to calculate the COI. The COI algorithm focuses on dividing the allocated space of each item by its turnover in a given period of time, and then ranking the items from the lowest to highest COI. Afterwards, following a space filling curve, the highest ranked COI items are assigned the easiest accessible and more convenient locations closest to the I/O. Kallina and Lynn (1976) present advantages and disadvantages of the COI, and practical rules on the implementation of the COI method.

The implementation of COI in warehousing has been very prevalent; however, the great disadvantage is that demand rates of products constantly change –in most of the cases due to products' seasonalities. Thus, COI would require several re-assignments on the picking floor, resulting in an increase of labor in stock reprofiling. Moreover, in ordinary DC's where order picking is performed in a sequenced manner, as it is in a batch picking processes, the COI looses

flexibility in assigning products that appear in the same picking route close to each other. Thus, the risk of assigning SKU's next to each other when the products are not related in the picking route is very high, leading to an increase in travel. Therefore, product correlation should be considered in warehouse product assignment policies.

2.1.3.2 Correlated Assignment

In warehousing, *product correlation* can be described as the set of products that are often requested/ ordered together. For example, customers tend to purchase a toothbrush together with toothpaste, or flashlights with batteries. In this case, it would be valuable to store correlated products close to each other to reduce traveling distances in the order picking process.

To apply product correlation, Frazelle and Sharp (1989) present a statistical procedure that identifies pairs of products that are correlated and should be stored together. They perform a simulation study on a one-aisle Miniload AS/RS to compare random storage and correlated assignment, and show that correlated assignments can reduce by 30-40% the number of required retrievals.

The limitations of correlated assignment are quite challenging. It requires readily available data about the relationship between items which might not be readily available in a warehouse with 10,000 different items – nearly 50 million product pairs. Also, product correlation might have different characteristics such as: safety issues (i.e. flammability), product fragility, shape, weight, etc., which constraints the decision of location assignment.

2.1.3.3 Quadratic Assignment Problem

The problem of assigning locations to a set of indivisible facilities was first modeled by Koopmans and Beckmann (1957) as a quadratic assignment problem (QAP). In warehousing application, one can think of products as facilities, and storage locations as locations. The QAP model considers facilities to be of equal shape and the distance is measured from centroid to centroid. It then locates facilities based on the flow between them, with the objective of reducing the total traveling distance. The objective function results in a second degree function of the variables (from there the name of quadratic), see equation 2.1. Montreuil (1990) formulated the facility layout problem using Mixed Integer Programming, with the difference from the original QAP that the facility is allowed to take irregular forms and the distance from facility to facility is measured from their respective I/O points instead of their centroids.

Quadratic Assignment Problem formulated as a non-linear programming model:

$$Min\ z = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} f_{ik} d_{jl} X_{ij} X_{kl} \qquad (2.1)$$

s.t.

$$\sum_{j=1}^{n} X_{ij} = 1 \quad ; \forall i \qquad (2.2)$$

$$\sum_{i=1}^{n} X_{ij} = 1 \quad ; \forall j \qquad (2.3)$$

$$X_{ij} \in \{0,1\} \quad ; \forall i, j \qquad (2.4)$$

Equation 2.1 minimizes the total traveled distance *d* between locations *j* and *l*, multiplied by the frequency of flow *f* between facilities *i* and *k*. Constraints 2.2 and 2.3 restrict locations to be assigned to more than one facility and facilities from occupying more than one location.

Finally, constraint 2.4 defines the decision variable *X* as binary: 1 if facility *i* is assigned location *j*; 0, otherwise.

The QAP is a well known NP-Hard combinatorial facility layout problem where the computation time increases exponentially as the problem size enlarges, making it unrealistic to solve large scale problems. Another approach to the facility layout problem is through space filling curves. Targeted to production facilities, Bozer et al. (1994) introduce the use of spacefilling curves in the facility layout to exchange departments in more powerful routines than two-way or three-way exchanges. However, despite the improvements in algorithms developed in the past, to date, there is no algorithm that can solve the QAP in polynomial time; thus, the large-scale application of the QAP only exists in the literature and theory.

Furthermore, similar to *product correlation*, the QAP only considers weights between pairs of products, which would be meaningless in product assignment of a fast picking area where the picking routing contains multiple items. Approaches dealing with location assignment in sequenced picking are presented next.

2.1.3.4 Order Orienting Slotting

Often in DC's, order pickers follow a route to pick several items belonging to an order, just like shoppers would do in a grocery store, in which the traveling distance relies on the picking list content and the routing strategy used. Thus, it is of great relevance to analyze the picking sequencing when assigning products to storage locations in a warehouse. Following this idea, Heragu et al. (2007) proposed an assignment strategy called *Order Oriented Slotting* (OOS). The concept of OOS integrates the picking frequency between items with their single popularity to assign them a storage location in the warehouse such that the total travel distance

needed to pick all the orders is minimized. Different from COI which assigns locations to items based on their frequency and stock size, OOS stores in a sequenced manner sets of high flow items that appear together within a large number of orders. Under the OOS approach, order pickers can pick many products belonging to an order in nearby locations without having to walk idle to reach the next item in the sequence. Based on the QAP, the OOS was formulated as an integer linear programming model when the routing is S-shaped and for Vertical Lift Model (VLM). Due to the complexity of the model when solving large scale warehousing problems, several heuristics were presented which delivered results within 6% of optimality.

The OOS strategy proposes a promising approach to product assignment. Nevertheless, the OOS model possesses a great disadvantage. It does not work well when the picking process is batched. OOS would only be efficient when every SKU is picked in a fixed sequence among other items, meaning that a SKU should not appear in different picking sequences. In a batching scenario, this is not the case. SKUs may appear in multiple sequences with different items to be picked in different route lengths. It should not be expected that customers would order the same set of products all the time.

2.1.3.5 Forward-Reserve Problem

As discussed in Chapter 1, many DC's are configured in a *forward area* for broken-case and full-picking and a *reserve area* for pallet picking and stock storage, in order to minimize the total costs of picking and replenishing. Bozer (1985) considers the problem of splitting a pallet rack into an upper reserve area and the lower as a forward area. He also shows when a separate reserve area is justified.

The decision of what items to place in the *forward* or fast picking area and in what quantities is known as the *forward-reserve problem* (FRP), first modeled by Hackman and Rosenblatt (1990). They present a heuristic that uses a ratio called *Economic Assignment Quotient* (EAQ) to solve the assignment-allocation problem. The EAQ can be described as the ratio of the annual number of request *R* for item *i* and the square root of its annual demand *D*, shown in equation 2.5.

$$\frac{R_i}{\sqrt{D_i}} \qquad (2.5)$$

The heuristic ranks the product based on their EAQ –highest to lowest- and assigns optimal quantities to the forward area for each product until it is full. Hackman and Platzman (1990) develop a mathematical programming model that solves more general instances of the Hackman-Rosenblatt model. Frazelle and Hackman (1994) formulate and develop two algorithms to solve the FRP, also incorporating congestion constraints and parameters being dependant on the storage volume. They prove that items may be ordered *a priori* according to the EAQ which depends on the characteristics of the items and is independent of the warehouse parameters. The algorithms determine the size of the forward area along with allocating items. They conclude that the costs of picking and replenishment of the forward area depends on the size of the forward area. Lately, Bartholdi and Hackman (2008) formulate it the *labor efficiency ratio* based on the EAQ.

$$\text{Labor Efficiency} = \frac{P_i\ (number\ of\ picks\ of\ sku\ i)}{\sqrt{f_i}\ (cartons\ of\ sku\ i)} \qquad (2.6)$$

The ratio represents the marginal net benefit measured in total labor –picking plus replenishment- of assigning an item into the forward area. They present an analysis of the net benefit obtained by allocating products based on their *labor efficiency*, and demonstrate that greater benefits can be obtained over using traditional methods such as Equal Time Allocation and Equal Space Allocation. They also compare *labor efficiency* to COI, and argue that the COI fails to consider restocking costs in the assignment of items, and therefore, is just a measure of *picking efficiency*, but not of *labor efficiency*.

The FRP is an effective model in the allocation problem, but not so much in the assignment problem because it does not analyzes the most convenient locations for each item, neither does it includes the flow relationship between items and picking sequences.

Other disadvantages related to the FRP are that in practice DC's or any retail store would generally allocate all of its products in the forward fast picking area and leave only the reserve area as storage space to be used for replenishing the picking area. This would literally give little value to the problem of deciding which items to put in the fast picking area. In addition, typical storage of products in the fast picking area happens in predefined slot sizes given by the shelve-configuration of the storage racks, thus, the amount of space to allocate for each item becomes trivial.

2.2 Sequencing Algorithms

In an era of customized orders, it is of importance to enable assignment strategies to consider the existence of products in different picking sequences, as it would appear in a batching process. Some of the related research in order retrieval sequencing found in the warehousing literature is presented next.

2.2.1 Order Retrieval Sequencing

Bartholdi and Platzman (1986) analyze several algorithms that sequence the retrieval of items from a carousel conveyor. They conclude that the appropriate choice of heuristic for retrieval depends on both the order-rate and the item-density. As these increase, simpler greedy heuristics are recommended to sequence the retrievals. In a similar work, Van den Berg (1996) presents an efficient dynamic programming algorithm to sequence the retrieval of a set of orders in a single carousel. The problem is simplified to a Traveling Salesman Problem (TSP) and solved to optimality.

Following retrieval strategies, Hwang and Song (1993) present a heuristic procedure to solve the problem of sequencing a given set of retrieval requests on a man-on-board storage and retrieval warehousing system. The problem is studied as a TSP to obtain the optimal tour for the subset of orders. They also develop the expected travel time models based on a probabilistic analysis for single and dual commands assuming a randomized assignment policy. Elsayed et al. (1993) study the sequencing and batching procedures in an automated storage/retrieval system (AS/RS) for minimizing earliness and tardiness penalties in order retrievals. An algorithm is

developed to rank the orders for assigning them to batches such that the time of completion of every batch minimizes the penalty of earliness or tardiness.

As we can see, most of the literature related to sequencing of orders in a warehousing is focused on retrieval algorithms were the items' storage locations are known *a priori*, if not, in most cases, the storage locations are randomly assigned. Although not mentioned in the literature, we can proof by generalization, that if any retrieval strategy depends upon the location assigned to items, then, the location assigned to items possesses a significant impact in the retrieval performance, and therefore, an optimal storage assignment will certainly speed the retrieval process.

2.2.2 Linear Placement Problem

The storage location assignment in sequenced picking can be thought as the *Linear Placement Problem* (LPP). The LPP is a complex non-polynomial (*NP*) combinatorial problem, where all the possible ways of placing n items is *n-factorial* ($n!$).

The LPP has had a significant application in the design of information systems, such as data mining and electronic integrated circuits. In information systems, Morse (1971) concentrates in the optimal assignment of books in a bookstore such that the items more closely connected in content are close together, in order that a person looking for those items focuses his search on the smallest part of the store as possible. The degree of connectedness between two items i and j is expressed in terms of a correlation index. Several approximation equations are presented to estimate the correlation indices. Once the correlation indices are known for each pair of items, a position x_i is assigned for each item i along a linear classification scale which indicates the optimal ordering of books in the shelves.

Saab and Chen (1994) present an effective heuristic algorithm to the LPP in the design of integrated circuits. The objective is to minimize the length of wire used to connect a set of nodes. In the placement search of all nodes, the algorithm begins with simple moves and gradually shifts towards more complex moves.

Applying LPP to warehousing, the problem objective would be to decide which locations should be assigned to items such that the distance between items that come in the same picking order is minimized. In a recent work, Wutthisirisart (2008) constructs a greedy-heuristic algorithm which compares all the sequenced order picks and assigns to the most convenient location the items in the sequence that minimizes the delay of the rest of the sequences. The concept of delay is defined as the number of locations that the order picker would need to travel idle, where no picking is done. After a sequence has been assigned, the items contained in such sequence are eliminated from the rest of the order list, and the search for minimum delay is repeated until all items have been assigned. The result of the algorithm comes as a linear ranking of items, which describes which items should be assigned to the first location before others. Although the heuristic runs in polynomial time, it is advantageous when the number of sequences p is smaller than the total number of items n to be assigned.

2.3 Replenishment Strategies

In chapter 1, the importance of replenishment in warehousing was discussed. However, the research done in the replenishment area is still in its early stages, pointing out that further work should be done in this field.

Van den Berg et al. (1995) formulate a binary programming model to find an assignment of unit-loads to the forward area that minimizes the expected labor-time during the picking period. One of their assumptions is that the order picker performs concurrent picking and replenishment operations during the same period. Under such an assumption, a second model is derived which adds a replenishment constraint that can be attributed to either possible congestion or staffing limits, in order to envision situations were concurrent replenishment during the picking period may be limited. They compare the heuristics to popular methods in practice and show that significant savings can be obtained.

Kim et al. (2003) present a tote replenishment logic to minimize the set-up time between picking cycles for the efficient operation of a warehouse in short cycle time environment. The set-up time is defined as the time to replenish the totes that contain the products to be retrieved in the next picking cycle. Lastly, a negotiation-based algorithm is proposed to determine the storage location for each product.

Hollingsworth (2003) focused on minimizing replenishment costs through a dock-to-forward (DTF) strategy, which bypasses reserve storage to reduce additional material handling. The DTF strategy transfers the full pallets received at the dock door to the forward picking area. Several DTF strategies were tested via simulation, where the best-performing DTF strategy reduced replenishment trips by as much as 24% over a system with no DTF.

Jernigan (2004) considered multi-tier inventory systems to minimize the total cost of picking items and replenishing storage locations. A multi-tier inventory system involves spreading inventory amounts closer to the order pickers; as would be a picking shelf that is replenished from a nearby storage location, and the location is replenished from bulk storage, forming a three-tier inventory system. A heuristic is developed to determine the storage areas to which to assign items and the quantities in which to store them in each time period, while minimizing the costs of picking, costs of space dedicated to restocking storage locations, and reassigning the items over multiple periods.

2.4 Summary of Literature

Most of the SKU assignment methods described in the literature formulate the assignment problem without considering the sequencing structure of order pickers in ordinary fast pick areas.

A critical observation of assignment methods such as COI and Labor Efficiency reveals that they are not a logical way of slotting because they assume that a pick tour consists of a single SKU. But when a typical order contains multiple items and all of these must be picked in one tour, COI and Labor Efficiency loose the correlation of items that are picked together; so the risks of assigning SKUs next to each other when the products are not related in the picking route is very high, leading to an increase of travel for order pickers.

More robust assignment methods as QAP and Correlated Assignment do consider the correlation of SKUs, but they do not capture their correlation with their respective sequences.

Furthermore, in OOS, although the SKU assignment is done with respect to its order route, it does not work well when SKUs are picked in different orders.

As seen in the literature, surprisingly, the storage assignment of items in manual sequenced order-picking is relatively new within the warehousing application. In addition, replenishment has not been integrated in the SKU assignment problem; instead, it has been formulated separately to minimize the number of restocks and the traveling involved from bulk warehouse to the fast pick area. Thus, the motivation of this research is to present a SKU assignment model that is able to minimize the walking distances of order pickers and restockers in a fast picking area, while at the same time capturing the sequencing factors of order picking processes, which is the nature of typical order picking in DCs.

CHAPTER 3

METHODOLOGY

This chapter first introduces the problem structure for a fast picking area to be analyzed and solved. Second, an integrated binary mixed integer linear programming model is formulated to optimize the total traveling distance by order pickers and restockers. Finally, a set of heuristics are proposed to solve large scale problems in a reasonable amount of time.

3.1 Problem Structure

As mentioned in Chapter 1, the fast picking area warehouse configuration chosen for this research consists of a manual picker-to-item environment with multiple items per order. The order picking and replenishment structures to be considered are presented next.

3.1.1 Order Picking Structure

3.1.1.1 Fast Pick Area Configuration

In a fast picking area, the order picking process under study requires pickers to enter the aisle from the I/O point and follow an S-shaped route to pick the set of items that belong to a certain order. During the picking process, order pickers travel through the aisles with a cart containing the box where the items are placed to fulfill the order. For simplicity of analysis, it is

assumed that an order can be fulfilled in a single box. Once an order is fulfilled, the box is dropped on a belt-conveyor and sent to the shipping area. At this point, the order picker returns to the I/O point to start picking a new order. See Figures 3.1 and 3.2.

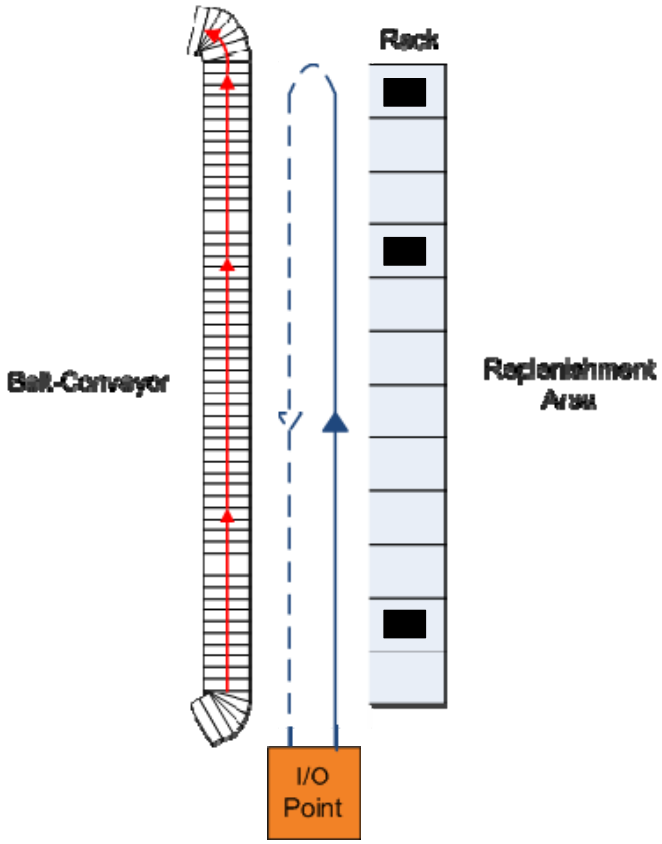


Figure 3.1 Single-aisle picking structure. The arrows represent the traveling direction of the order pickers; a dashed arrow is a return to the I/O point after completing an order. The black squares are the SKUs to be retrieved from the rack.

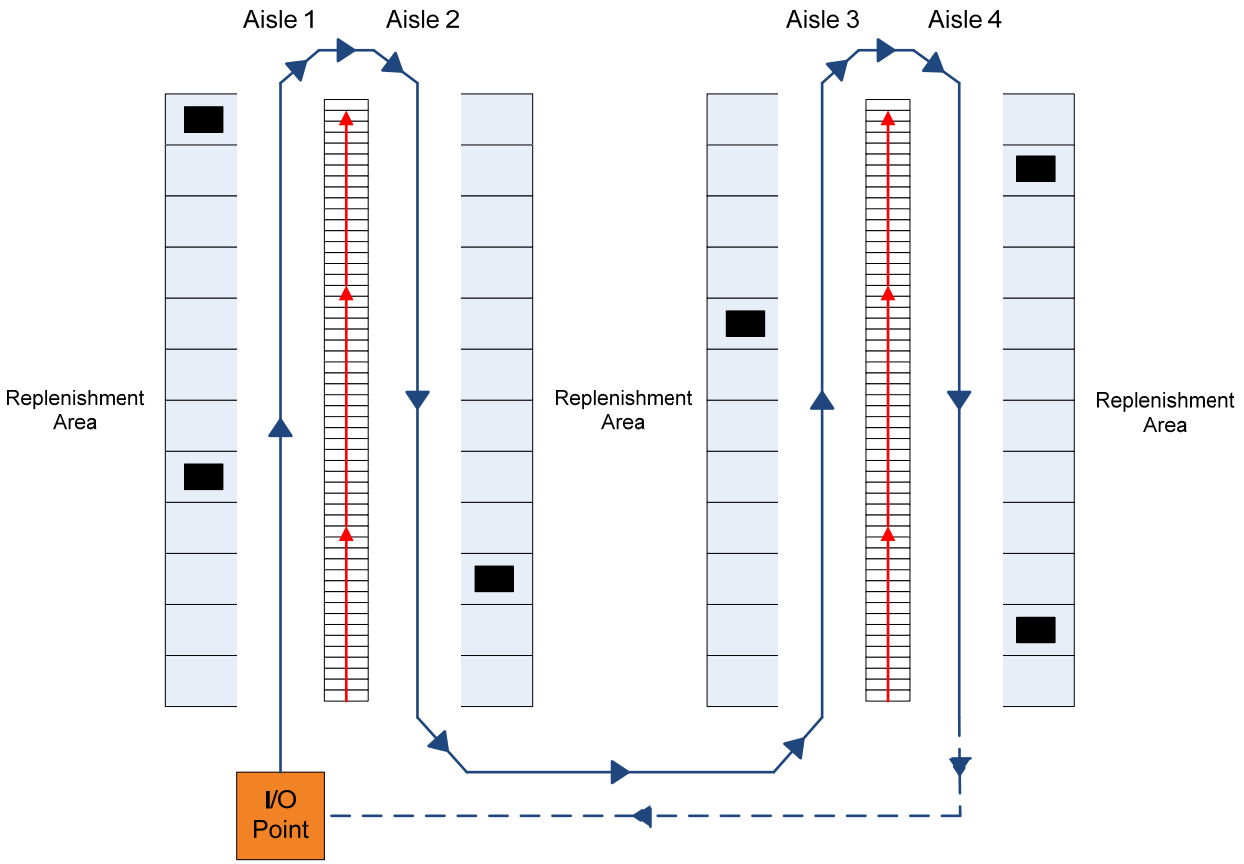
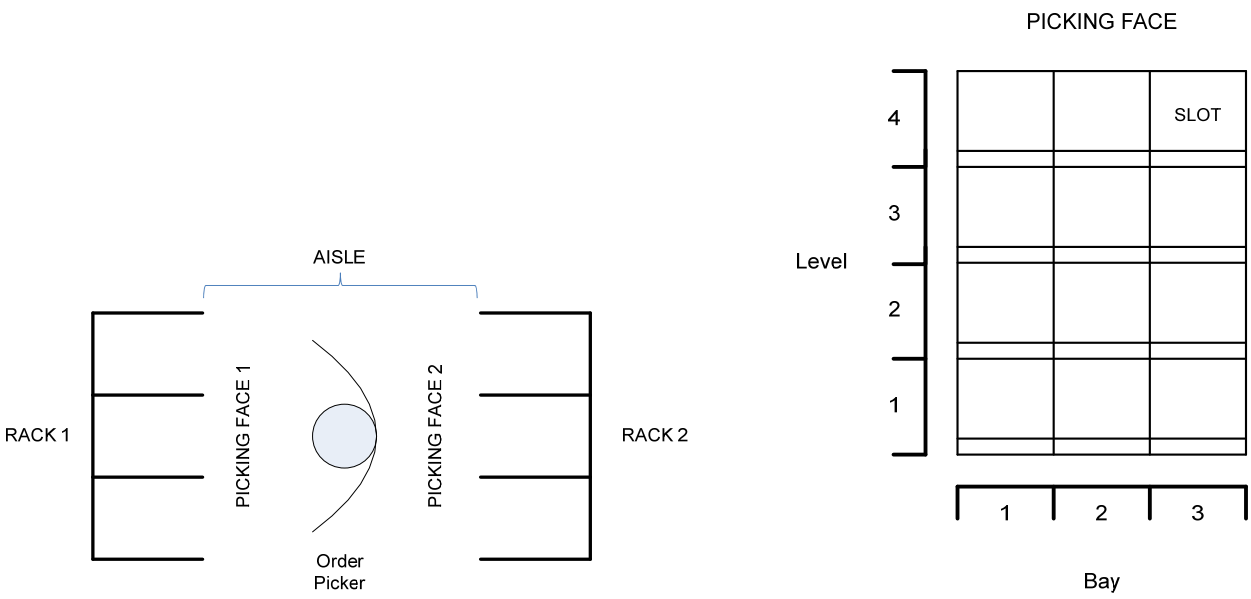


Figure 3.2 Multi-aisle picking structure following an S-shaped routing policy.

It should be pointed out that for the problem structure the return distances to the I/O point are ignored. This is because the purpose of this research is to minimize the traveling distances required to fulfill the orders, thus, it does not concern distances after picking. However, it is expected that a minimum picking travel will result in a minimum return travel as well.

The products are located in gravity-racks. Every rack contains bays with several levels of shelves where products are slotted. Since the use of gravity-racks requires products to be retrieved from the front of the rack, we will designate such access as the *picking-face*. Figure 3.2 illustrates the *picking-face* configuration.



a) Top view of a picking- face.

b) Front view of a picking-face.

Figure 3.3 Views of a rack configuration within a picking aisle.

Although in practice a bay would contain multiple SKUs, for the study of this research it is assumed that each bay contains cartons of the same SKU to allow slotting flexibility and avoid dealing with explicitly ergonomic issues. Another assumption is that the same SKU number cannot occupy more than a bay location in the entire fast pick area, thus, every SKU must have a single dedicated Bay location.

3.1.1.2 Order Picking Sequencing

As mentioned earlier, an order is defined as the set of products that need to be picked together from the racks to fulfill the order. To gain a better understanding of how orders flow in the fast picking area, the fulfillment process is analyzed from the order perspective instead of the picker perspective. By doing so, operational factors such as: order batching, batch size, required number of tours, etc, that affect order pickers, do not need to be considered. In this way, the tour

or segment each individual order travels to be fulfilled will be called order picking sequence. To avoid confusion, the order picking sequencing does not mean that the extraction of products is done in a restrictive sequential manner; it only describes the items that are picked together in a segment, and is called sequencing because one pick is done after another.

In a DC, although customer orders tend to be customized, it is possible to see through time that several orders repeat the same product mix, thus, the same picking sequence. This can be expressed as orders that follow the same picking sequence at a certain frequency.

Order picking sequences can be generated for determined periods of time by means of forecasting or extraction of historical shipping data. Although in real practice an order might take several days to get fulfilled, for our study it is assumed that an order has the ability to be fulfilled in one day.

An example of a set order picking sequences is shown in Table 1. From left to right: the first column is the order picking sequence number; the second column shows the Bay numbers to be visited to retrieve the SKUs for that order picking sequence; and the third column indicates the frequency in which orders repeat the sequence (product mix). For example, the first sequence needs to visit Bays 1 and 3 six times, meaning that there are six orders that contain items in Bays 1 and 3; the second sequence visits Bays 1, 2, and 4 four times; and so on.

Table 3.1 Example of an Order Picking List.

Sequence	SKU numbers / Bay Number					Frequency
1	1	3				6
2	1	2	4			4
3	5	6	7	8	9	4
4	2	8				4
5	7	9				3
6	4	5	6			3
7	2	6	7	8	9	1
8	1	2	7	8	9	1

3.1.2 Replenishment Structure

3.1.2.1 Replenishment of the Fast Pick Area

For our study, restockers and order pickers are two different staff members, and so replenishment occurs in advance before order picking takes place.

Figure 3.4 illustrates the replenishment structure in a single-aisle fast picking area. Arriving from the bulk storage area, the SKU cartons to replenish the fast picking areas are staged at the I/O point. Restockers are then required to put away the SKU cartons into their respective slots and, afterwards, return to the I/O point for more cartons to replenish. Opposite to order picking, the replenishment process occurs at the back of the picking-face.

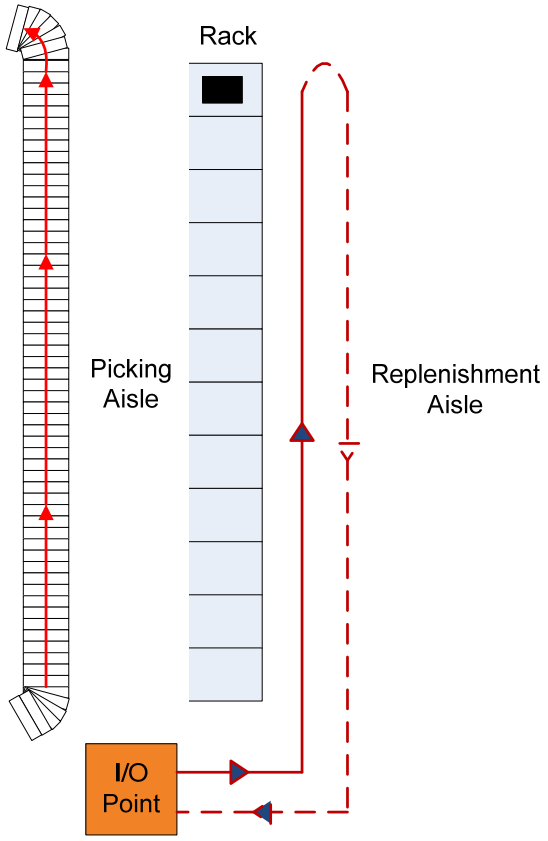


Figure 3.4 Replenishment structure. The black square is the SKU to be replenished.

The restockers are assumed to carry only one carton at a time from the I/O point to its respective slot location. Return distances to the I/O point are ignored.

In a multi-aisle fast picking area, the replenishment travel is similar to that of the order pickers, illustrated in Figure 3.2, except that the walking is done behind the picking-faces.

3.1.2.2 Restocking Requirements

The restocking requirements are generally a function of: carton volume, pick flow and safety stock. The carton volume v indicates the standard quantity of units contained in the SKU carton. The pick flow p is the total number of units of each SKU retrieved during the picking period, independently of the number of times the SKU location was visited. Knowing the pick

requirements of each SKU and its carton volume, the minimum number of cartons R required to replenish the fast pick area can be easily obtained by $R = p/ v$. Because the smallest replenishment unit is one case, any results with decimals have to be rounded up to the nearest integer.

Safety stock is a common practice in DCs to absorb fluctuations in demand; however, for this research it is assumed that no safety stock is necessary for any of the SKUs.

To illustrate how the restocking requirements work, Table 3.2 presents an example of the restocking requirements based on the order picking list of Table 3.1.

Table 3.2 Example of Restocking Requirements.

SKU Number	Volume (v) (units/carton)	Pick Flow (p)	Cartons	Cartons Required to Replenish (R)
1	10	22	2.2	3
2	8	40	5	5
3	5	12	2.4	3
4	6	21	3.5	4
5	2	7	3.5	4
6	4	40	10	10
7	3	18	6	6
8	7	10	1.4	2
9	6	27	4.5	5

3.2 Problem Analysis

Having described the problem structure, the purpose of this research is to re-assign the Bays of the fast picking area to better locations such that the segment lengths of the order picking sequences and the traveling distance incurred by restockers are minimized.

The decision to focus on the reduction of travel lengths for order picking sequences is based on the assumption that if the product mix belonging to a certain order is assigned close to the I/O point and kept closed to each other, then orders will require little travel to be completed; and order pickers will travel less to fulfill orders. Even more, under this approach, subsequent operations, like order batching, become more efficient, i.e., batching orders according to their picking sequence segments increases the completion of multiple similar orders at minimum travel distance.

3.2.1 Analysis of Order Picking Sequence

The special structure of S-Shapes routes enables us to view the distances traveled by order picking sequences from a linear perspective. Figure 3.5 illustrates the travel of each order sequence previously presented in Table 3.1 –the preliminary assignment of locations to SKUs was done in increasing order according to its number value. In Figure 3.5, it can be observed that all of the orders have some idle travel, meaning that they are crossing bays where no picking is needed. In an ideal warehousing world, we would want every order to only travel bays where picks are required, but because it is not physically possible to overlap one bay over another there will always be some idle walk involved. During this study, we will refer to the length of an order sequence as the number of bays it has traveled until fulfillment.

The total distance traveled by the set of orders can be observed at the right of Figure 3.5. The amount of distance *D* traveled by each order sequence is given by the product of its length *L* (the number of bays transited) times its frequency *F* (the number of orders containing the same product mix). It becomes clear that to minimize the total distance *D* it is necessary to reduce the length *L* of order picking sequences. The achievement of such a minimization is the core contribution of this research.

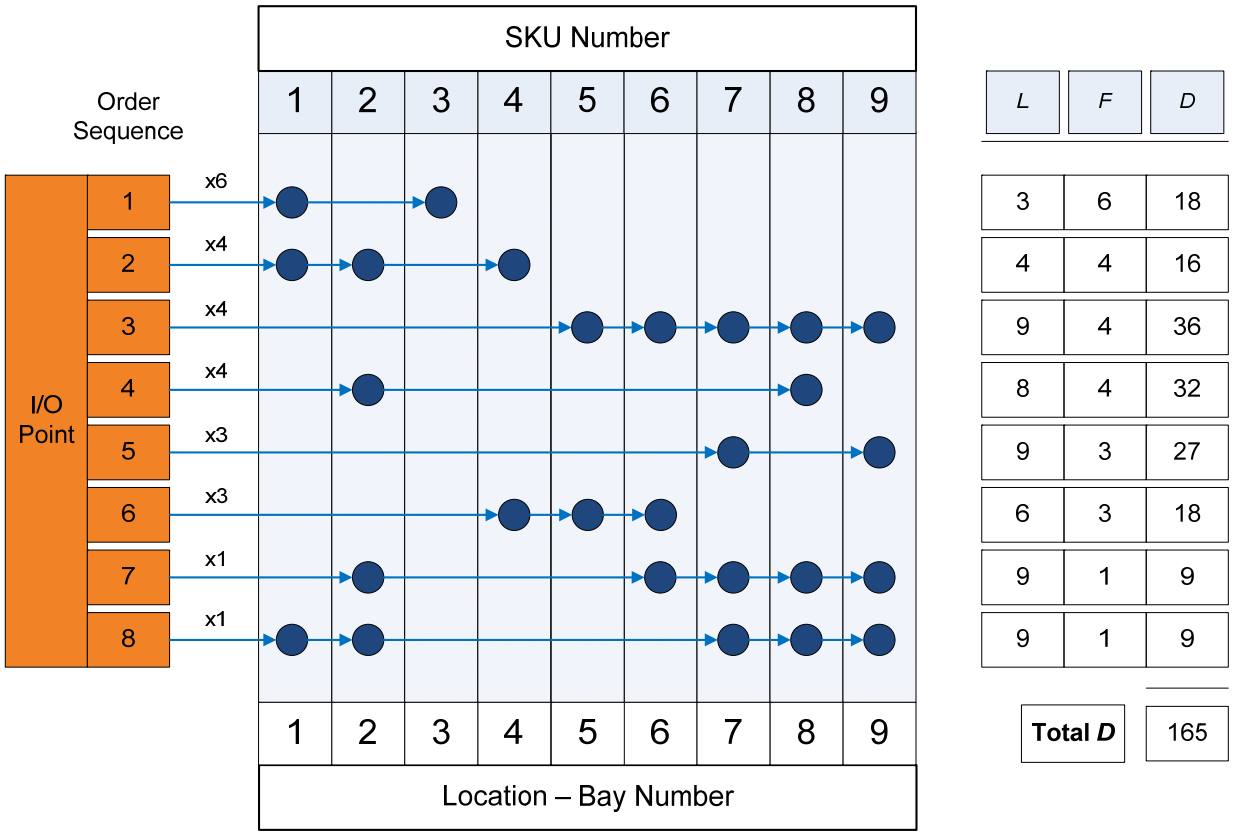


Figure 3.5 Representation of the traveling path done by each order picking sequence and the amount of distance involved.

To reduce the number of bays transited by all orders it is necessary to swap or move around the SKU locations. This leads to a combinatorial problem, where the total possible

combinations in which *n* SKUs can be arranged is *n*!. The example given in Table 3.1 contains 9 SKUs, thus, there are 9! = 362,880 ways of assigning SKUs to bay locations! It will become very time consuming to list all the possible combinations to figure out the one that minimizes the total distance, and almost unrealistic to list when the number of SKUs increases.

According to the literature review discussed in Chapter 2, there have been several methods used in the assignment problem. If we were to solve this problem using the *Cube-per-Order-Index* (CPO) approach, the methodology would concentrate on ranking the SKUs by their number of visits. Based on Table 3.1 all that would be needed to do is count how many times each SKU repeats in the order picking sequence list and sort them from highest to lowest. The highest SKUs are then assigned to the locations closest to the I/O point.

Although the CPO method is very practical and straight forward, it would only give good results if the retrieval of products occurred in a single SKU command basis, basically, reaching a SKU and coming back to the I/O point. However, the structure of the problem we are dealing with is different. The order route has multiple SKUs, and the return to the I/O point is not done until the last SKU of an order is retrieved. Therefore, applying CPO is not efficient since the assignment strategy will lose the correlation between SKUs belonging to the same order.

Another way to approach our problem is by using methods that consider SKU correlation such as the *Quadratic Assignment Problem* (QAP) or *Order Oriented Slotting* (OOS). The solution method would then reduce the distance that exists between pairs of SKUs that are frequently visited together. However, assigning SKUs to locations with respect to their frequency weights does not guarantee that the order sequence length is reduced as well because the information on which SKUs belongs to an order sequence is not considered.. Based on the fact that an order sequence is not completed until all of its SKUs are picked, even when distances

between pairs of highly correlated SKUs are minimized, the overall distance of an order sequence can potentially get worst. For example, an order sequence could extend its fulfillment because pairs of SKUs that do not belong to that order were assigned before SKUs that actually belong to the order. The fact that our problem contains SKUs that repeat in different order sequences at different frequencies downgrades the solution quality of correlated assignment strategies.

Finally, from a scheduling perspective, giving higher priority to the SKUs that belong to the most frequent order sequence sounds like an acceptable solution method. However, SKUs in larger sequences may become candidates to be assigned first at the expense of increasing the lengths of other sequences that are almost equally frequent. Thus, a more robust approach to this problem is necessary.

3.2.1.1 Solution Approach for Order Picking Sequencing

Analyzing Figure 3.5, it can be observed that the factors that determine the distance traveled by each order sequence are their length and frequency. The frequency of orders is a constant value that cannot be changed, thus, the only variable to this problem is the length of an order sequence. Because an order sequence is not fulfilled until all of its SKUs have been picked, it can be concluded that the length of an order sequence is a function of the bay number in which the last SKU in the order list is located. For example, sequence 1 has SKUs 1 and 3, located in bays 1 and 3, respectively; since the last pick is done at bay 3, then the length sequence 1 travels is equal to 3 distance units.

To express the distance of each order sequence the following notation will be used:

D_i = Bay location number of SKU i

S_k =Order sequence k

F_k = Frequency of orders for sequence k

L_k = Length of order sequence k

where,

$$L_k = [max\{D_i \in S_k\}][F_k] \quad (3.1)$$

Equation 3.1 sets the length *L* of a given order sequence *k* to be equal to the maximum bay location number among the set of SKUs *i* belonging to order sequence *k*, times its frequency *F*; where the maximum bay location of a set of SKUs represents the farthest location in which an SKU is assigned.

The minimization of the lengths of all order sequences is then be expressed as:

$$Min\ z = \sum_{k=1}[max\{D_i \in S_k\ ;\ i = 1,2,3, \dots, n\}][F_k] \quad (3.2)$$

So far, the distance traveled by order picking sequences have been analyzed, now, in the next section replenishment distances will be covered.

3.2.2 Analysis of Replenishment Flows

Compared to order picking sequences, the replenishment flow occurs in a simpler way. The amount of traveling distance *D* required to replenish each SKU depends on two factors: the bay number location *L* and the SKU turnover velocity *R*, which is a function of the carton volume capacity and the number of picks for each SKU. From the restocking data given in Table 3.2, the replenishment flow is visualized in Figure 3.6.

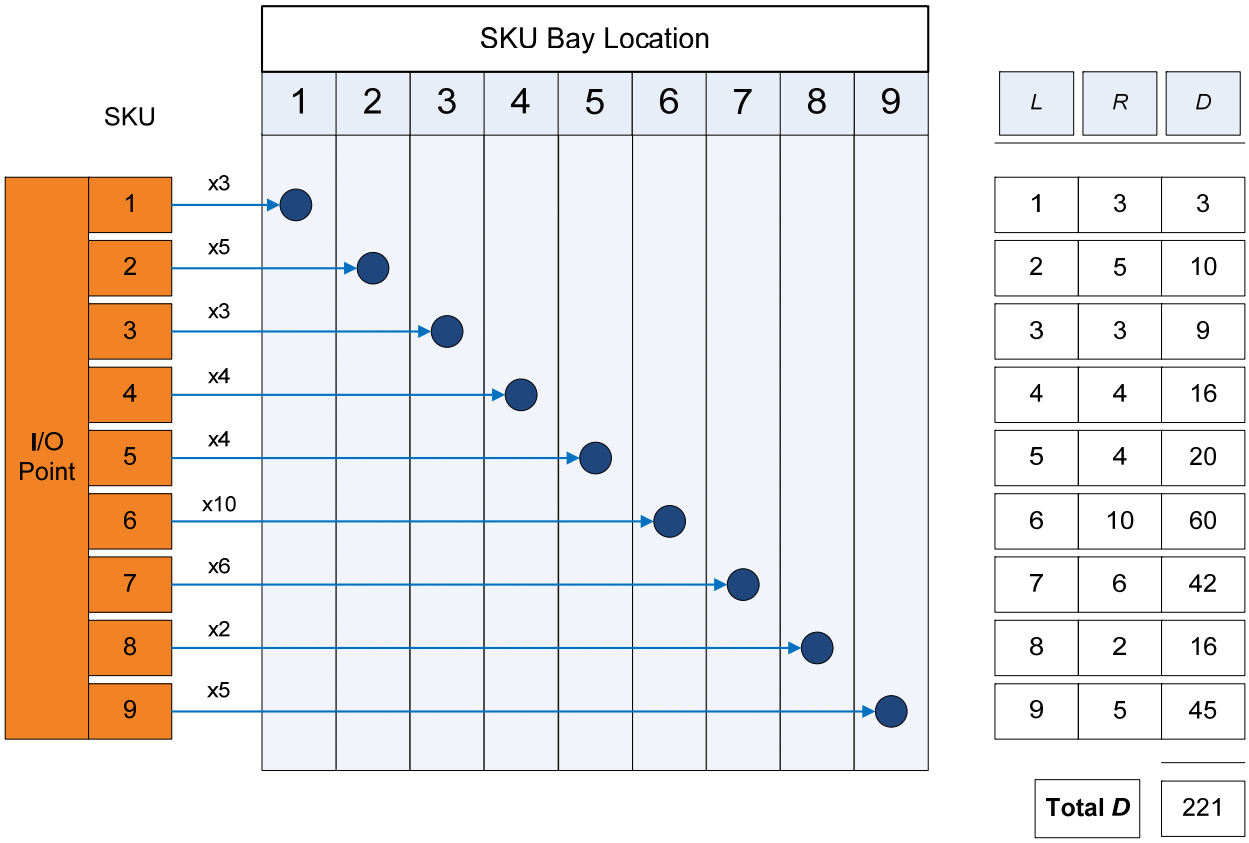


Figure 3.6 Representation of the traveling path for the replenishment of SKUs.

The distance travel for restocking each SKU is equal to the number of cartons (turnover) to replenish times the bay location where the SKU is located. From a replenishment perspective, it becomes clear that in order to minimize the travel distance involved, it is necessary to reassign

the most replenished SKUs to bay locations closer to the I/O point, i.e., the highest replenished SKU is number 6, with 10 cartons, however it is not the closest SKU to the I/O point based on picking.

Table 3.3 summarizes the traveling distances involved in order picking and replenishment based on Figures 3.5 and 3.6.

Table 3.3 Traveling distance results for the exercise in Table 3.1

Distances		
Order Picking	Replenishment	Total
165	221	386

From a myopic view, without considering how SKUs behave in the order picking process, solving the replenishment problem is very straight forward. All there is to do is to rank the SKUs from lowest to highest number of restocks. Nevertheless, the objective of this research is to minimize both order picking and replenishment distances. And assigning SKUs to bays locations with respect to their restocking velocity could potentially increase the distances of order picking sequences. The integrated model which considers both order picking and replenishment flows is presented in the next section.

3.3 Model

3.3.1 Binary Mixed Integer Linear Programming Model

To formulate the Binary Mixed Integer Linear Programming Model (BMILP), let's introduce some additional notation:

D_i = Bay location number assigned to SKU $i = 1,2,\ldots,n$

S_k =Order sequence $k = 1,2,\ldots,K$

F_k = Frequency of orders for sequence $k = 1,2,\ldots,K$

L_k = Length of order sequence $k = 1,2,\ldots,K$

R_i = Number of cartons to replenish for SKU $i = 1,2,\ldots,n$

b = Bay location number $b = 1,2\ldots,B$

3.3.1.1 Decision Variable

As was seen in the previous section, the bottom line for distance minimization in order picking and replenishment relies on where the SKUs are assigned in the floor. In this way, the decision variable is defined as a binary variable:

$$X_{ib} = \begin{cases} 1, \text{ if SKU } i \text{ is assigned to bay number } b; \\ 0, \text{ otherwise.} \end{cases} \quad (3.3)$$

3.3.1.2 Objective Function and Constraints

Based on what we have learned in Section 3.2, the minimization of order picking sequences and replenishment can be expressed as follows:

$$Min\ z = \sum_{k=1}[max\{D_i \in S_k\ ;\ i = 1,2,3,\dots,n\}][F_k] + \sum_{i=1} D_i R_i \qquad (3.4)$$

Equation 3.4 minimizes the total length distances within order picking sequences and the total replenishment distances. It can be observed that this equation does not have a linear programming structure. The linearization of the equation above can be achieved by setting $L_k = max\{D_i \in S_k\}$, according to equation 3.1, and moving $max\{D_i \in S_k\}$ to the constraints. The linearization yields the following Binary Mixed Integer Linear Programming (BMILP) formulation:

$$Min\ z = \alpha \sum_{k=1} L_k F_k + (1-\alpha) \sum_{i=1} D_i R_i \qquad (3.5)$$

subject to,

$$\sum_{b=1} X_{ib} = 1 \quad ;\ \forall\ i \qquad (3.6)$$

$$\sum_{i=1} X_{ib} = 1 \quad ;\ \forall\ b \qquad (3.7)$$

$$\sum_{b=1} bX_{ib} = D_i \quad ;\ \forall\ i \qquad (3.8)$$

$$D_i \leq L_k \quad ;\ \forall\ i \in k \qquad (3.9)$$

$$X \in \{0,1\} \quad ;\ \forall\ i, b \qquad (3.10)$$

The objective function (3.5) expresses in a linear form the minimization of the sum of distances of order picking and those of replenishment. The first part of the equation sums the total length distances of all order picking sequences. The second part adds the distances of each replenishment flow for every SKU. As learned in section 3.2, the replenishment flow distance of an SKU is equal to the bay location *D* where the SKU is assigned times the number of cartons *R* that flow to that location (replenishment distance = *D* x *R*).

Notice that the equation contains a positive constant α that provides relative weight to the objective function in assigning SKUs in a way that it benefits more order picking or replenishment. In real DCs, it is common to see that order picking and replenishment are not weighted the same because of distinct policies, i.e., labor cost rates, where the assignment of SKUs is done to exclusively benefit order picking at the expense of replenishment because the total labor costs of order picking is greater than those of replenishment. Depending on what is of more interest for a DC, the value of α can be determined empirically between 0 and 1 so as to obtain the right weight of these two activities.

Constraint (3.6) avoids SKUs to be assigned to more than one bay location. Constraint (3.7) ensures that each bay location is a dedicated storage for an SKU, thus, a bay location cannot store more than one SKU. Constraint (3.8) sets the bay location number for all SKUs when the value of the decision variable holds true. Finally, constraint (3.9) is derived from the aforementioned expression $max\{D_i \in S_k\}$. In its linear form, it ensures that the length of the order picking sequence is given the highest bay location number of the set of SKUs belonging to that sequence.

3.3.2 Additional Constraints

In practice, the way order sequences are fulfilled and SKUs are replenished is governed by the different aspects associated with the variety of SKUs in a DC, such as weight, volume, form, etc. For instance, an order containing heavy and light SKUs would most likely be fulfilled by picking the heavy SKUs first and the light ones at the end to ensure that the products are not damaged. Crushable SKUs, i.e. bows, would be picked at the end to optimize the box content capacity. Or, for example, an order containing regular shaped SKUs and irregular shaped SKUs would have regular shaped SKUs picked first to better utilize the space inside the box. On the replenishment side, there might be SKUs that only get replenished in full-pallets and so is more convenient to have their storage locations at the ends of the restocking aisle for better accessibility. Or maybe a group of SKUs get restocked by the same pallet, so their storage locations need to be close to each other. Indeed, there are numerous constraints that limit the assignment of SKUs in the floor, out of which three main types of constraints are formulated next.

a) Order Sequencing Constraints

$$D_i = b \tag{3.11}$$

$$D_i < D_j \tag{3.12}$$

The first equation (3.11) dictates that a given SKU i has to be assigned to a certain bay number. This would be the case of light SKUs that need to be picked first or crushable SKUs that are to be picked last. The second equation (3.12) ensures that a given SKU i has to picked

before an SKU *j*, as would be the case of heavy versus light SKUs, and regular versus irregular shaped SKUs.

b) Replenishment Ranges

$$b_1 \leq D_i \leq b_2 \qquad (3.13)$$

Equation (3.13) indicates that a given SKU *i* needs to be stored within a certain range of bay locations. This applies when SKUs should be replenished within certain bays because of their carton size.

c) Grouping of SKUs

G = Group number of SKUs to cluster

E = Space between first and last SKU belonging to cluster G

$$max_G - min_G \leq E_G \quad ; \quad \forall \ G \qquad (3.14)$$

$$D_i \leq max_G \quad ; \quad \forall \ i \in G \qquad (3.15)$$

$$D_i \geq min_G \quad ; \quad \forall \ i \in G \qquad (3.16)$$

Equation (3.14) ensures that the distance between the farthest and shortest location of the set of SKUs in cluster G does not overpass the specified space constant E for such cluster.

Equation (3.15) and (3.16) define the farthest and shortest location, respectively, of the set of SKUs in the cluster group.

The set of additional constraints discussed in a), b), and c) can be incorporated in the BMIP model and solved to optimality.

3.3.3 BMILP Solver - LINGO

In order to solve the BMILP mathematical model described in section 3.3.1 the use of an optimization solver software is necessary. For our convenience, LINGO software was used. LINGO has the ability to model large systems by expressing a group of several very similar calculations or constraints into SETS. A set might be a list of products, employees, trucks, etc. Each member in the set may have one or more characteristics associated with it (e.g. weight, location, price/unit, income). In our model, the SETS are: SKUs, Bay locations, and Sequences. The decision variable X_{ib} would then be a derived set of SKUs and Bay locations because it consists of every possible combination of assigning a SKU to a Bay location.

While a SETS section describe the structure of the data for a particular class of problems, a DATA section provides the data to create a specific instance of this class of problem and allows isolating things that are likely to change. In terms of our problem, based on Table 3.1, the data we know is the SKU number, the Bay Location Numbers, the Order Picking Sequences and their respective Frequencies.

The third component of the LINGO model is composed of set looping functions, where the relationships among the various attributes are stated. The power of these functions comes from the ability to apply an operation to all members of a set using a single operation. For the BMILP, the set looping functions are used to express the objective function and constraints.

3.3.3.1 LINGO SETS Code

Having described the three basic components of a LINGO model, the BIMLP model in LINGO for the exercise given in Table 3.1 is presented below.

```
Sets:

SKU;
Location;
SKU_Location(SKU, Location): Xib;
SKU_Attributes(SKU): Di;
Sequence;
Sequence_Attributes(sequence): Lk, Fk;
Replenishment_Flow(SKU): Ri;

Endsets

Data:

SKU=@ole();
Location=@ole();
Sequence=@ole();
Fk=@ole();
Ri=@ole();

Enddata

a=0.5;

MIN=a*@sum(sequence(k):Lk(k)*Fk(k))+(1-a)*@sum(sku(i):Di(i)*Ri(i));

Order_Picking=@sum(sequence(k):Lk(k)*Fk(k));
Replenishment=@sum(sku(i):Di(i)*Ri(i));


Di(1) < Lk(1);
Di(3) < Lk(1);

Di(1) < Lk(2);
Di(2) < Lk(2);
Di(4) < Lk(2);

Di(5) < Lk(3);
Di(6) < Lk(3);
Di(7) < Lk(3);
Di(8) < Lk(3);
Di(9) < Lk(3);

Di(2) < Lk(4);
Di(8) < Lk(4);

Di(7) < Lk(5);
Di(9) < Lk(5);
```

Di(4) < Lk(6);
Di(5) < Lk(6);
Di(6) < Lk(6);

Di(2) < Lk(7);
Di(6) < Lk(7);
Di(7) < Lk(7);
Di(8) < Lk(7);
Di(9) < Lk(7);

Di(1) < Lk(8);
Di(2) < Lk(8);
Di(7) < Lk(8);
Di(8) < Lk(8);
Di(9) < Lk(8);

```
@for(sku(i):@sum(location(b):Xib(i,b))=1);
!A sku is assigned to only one location;

@for(location(b):@sum(sku(i):Xib(i,b))=1);
!A location is assigned to only one sku;

@for(sku(i):Di(i)=@sum(location(b):Xib(i,b)*b));
!Assign location number to sku i;

@FOR(sku_location(i,b):@BIN(Xib(i,b)));
@FOR(sku_attributes(i):@GIN(Di(i)));
@FOR(sequence(k):@GIN(Lk(k)));
```

3.3.3.2 LINGO Solver Status Window

When solving a model in LINGO, it displays a solver status window which is useful for monitoring the progress of the solver and the dimensions of the model. The solver status window for the exercise in Table 3.1 is presented in Figure 3.7.

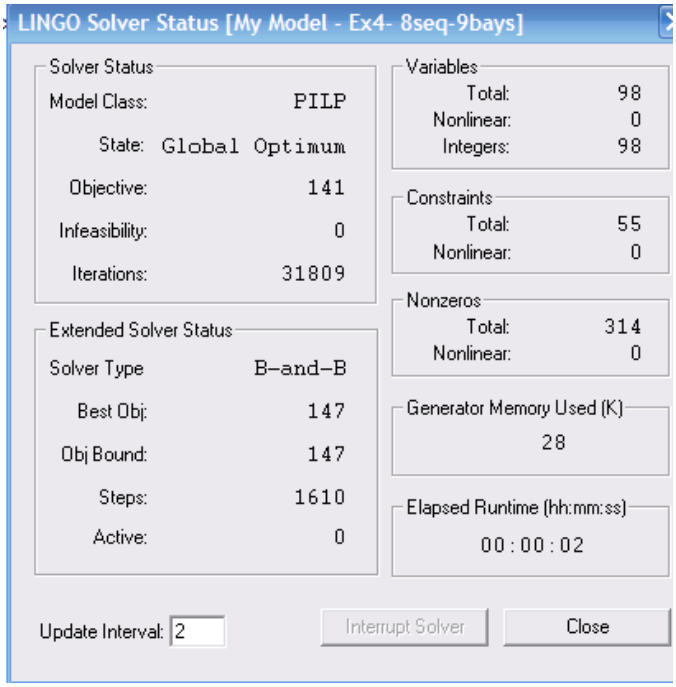


Figure 3.7 LINGO solver status window for exercise in Table 3.1.

Four important fields of the solver status window are discussed: *state*, *solver type*, *best objective*, and *objective bound*.

State. Gives the Status of the current solution. Possible states are "Global Optimum", "Local Optimum", "Feasible", "Infeasible", "Unbounded", "Interrupted", and "Undetermined". Once the solver can no longer find better solutions to the model, all optimized linear models will terminate in the global optimum state.

Solver-type. Since our model to be solved is an integer programming model, LINGO employs an optimization strategy called *Branch-and-Bound* (B-and-B). Branch-and-bound is a systematic method for efficiently exploring a solution space without having to enumerate all possible solution combinations.

Best objective. This field displays the best feasible objective value found so far.

Objective bound. This field displays the bound on the objective function. This bound is a limit on how far the solver will be able to improve the objective. At some point, the values of the *best objective* and *objective bound* may become very close. Given that the best objective value can never exceed the bound, the fact that these two values are close indicates that LINGO's current best solution is either the optimal solution, or very close to it.

3.3.4 BMILP Run Time Analysis

The BMILP model was successfully ran in LINGO. Further discussion in the solutions obtained for the example given in Table 3.1 and other case study scenarios are presented in the next Chapter. But, before continuing to Chapter 4 it is fundamental to analyze the extent of the problem sizes the BMILP model is capable to solve in an acceptable period of time.

A sensitivity run time analysis of the BMILP model, without considering the replenishment function, was performed over different problem sizes with respect to the number of SKU/Bays. Table 3.4 presents the different problem sizes used in this sensitivity analysis, and the running time required to solve them is illustrated in Figure 3.8. The lower part of Figure 3.8 illustrates the approximation of the best solution found to the lower bound found by LINGO.

It can be observed that the BMILP model solves to optimality small scale problems of less than 26 SKUs/Bays in seconds, but as the number of SKUs (Bays) increases in the problem structure the running time experiences an exponential growth. The largest problem size the BMILP model was able to solve consisted of 120 SKUs/Bays in 70 sequences, however, the solution did not improve after 14 minutes of running, and was only able to reach a feasible solution which was not a local or global optimal. The BMILP was also run for 120Bays in 100 sequences, but no solution was obtained in a reasonable amount of time. Furthermore, the

BMILP was not able to reach optimal results in short time for problems with more than 10 SKUs/Bays; therefore, the BMILP model can only be solved to optimality if the problem size is small. However, for problems with less than 120 Bays the quality of the solutions is satisfactory, falling within 20% of the lower bound and, consequently, close or equal to the optimal value.

The sensitivity run time analysis provides evidence that the BMILP model runs in Non-Polynomial time and, therefore, is not suitable for use to solve large scale problems where thousands of SKUs and hundreds of different order picking sequences are involved. For this reason, heuristics are needed to obtain approximate solutions. Simple heuristic approaches to solve this problem are presented in the next section.

Table 3.4 Problem sizes and their solutions used in the running time sensitivity analysis.

Number of Sequences	SKUs/ Bays	Run Time (min)	Lower Bound	Solution Found	Distance from LB
3	5	0.02	50	50	0%
6	6	0.02	78	78	0%
8	9	0.03	147	147	0%
20	26	0.82	1409	1415	0.4%
30	44	1.73	11449	12538	9.5%
60	65	3.30	3934	3991	1.4%
50	91	8.52	43262	51586	19.2%
70	120	13.55	122802	144500	17.7%

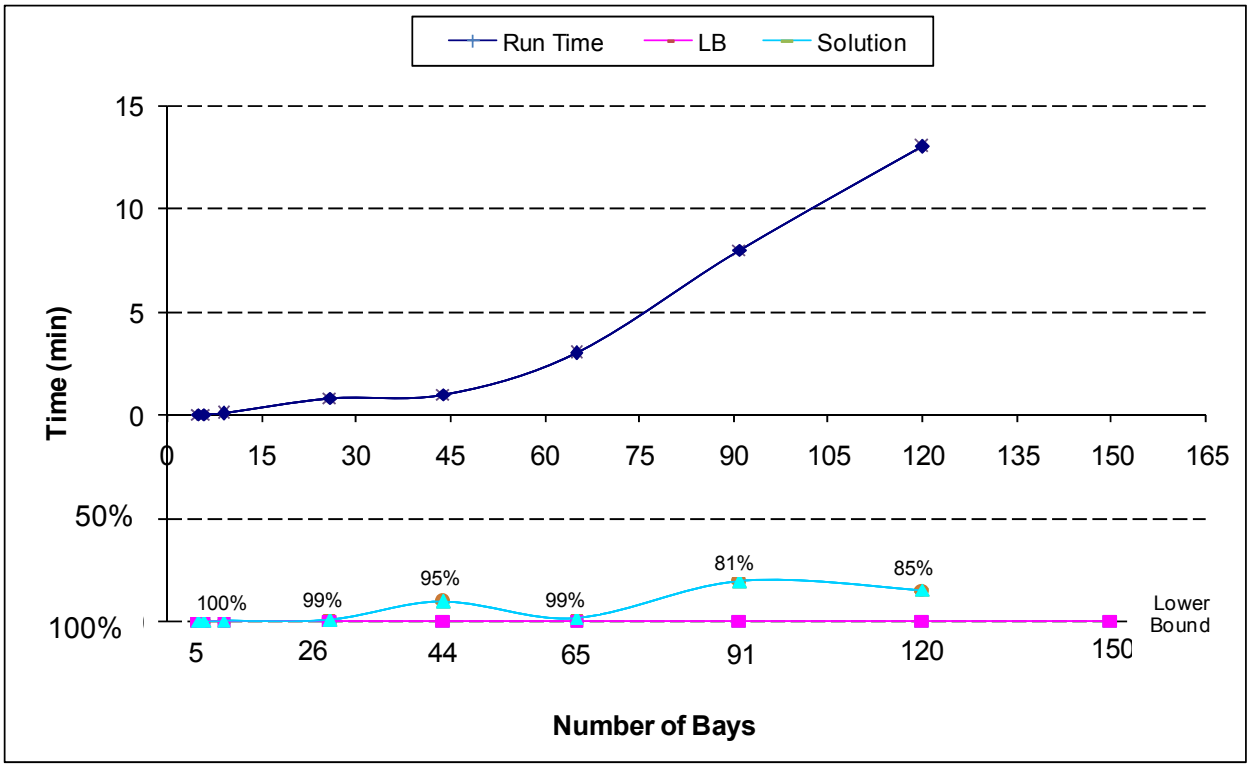


Figure 3.8 Graph of the running time of the BMILP model for distinct problem sizes and the quality of solutions with respect to the lower bound.

3.4 Heuristics for the SKU Assignment Problem in a Fast Picking Area

Five heuristics are presented in increasing degree of complexity for the general problem of SKU assignment in a fast picking area.

1. *Random assignment heuristic.* This heuristic randomly assigns SKUs to Bay locations.
2. *SKU popularity heuristic*. This heuristic uses the COI concept. First, count how many orders contain each SKU in the list of order picking sequences and sort them in decreasing order of their popularity. Then assign the SKUs in that same order to the Bay locations closest to the I/O point.
3. *Sequence popularity heuristic*. First, sort the order picking sequences in decreasing order with respect to their frequency, and then by the number of SKUs contained in each sequence. This way, highly frequent small sequences get assigned first. Second, assign the SKUs contained in the highest ranked sequence closest to the I/O point based on their SKU popularity. If there is a tie in their SKU popularity, the tie-breaker becomes the number of times the SKUs appears in other sequences.
4. *Maximum Dead-Walk Algorithm.* The concept of maximum dead-walk is to weight each order picking sequence by the longest dead-walk it could possibly have. The main idea behind it is to avoid having order picking sequences with a significant amount of idle walk in between.

   The following provides the steps of this heuristic:

   Step 1. Calculate the maximum dead-walk of each sequence by subtracting the number of SKUs contained in each sequence from the total number of

unassigned Bays on floor, then multiple the result by the frequency of the sequence.

Step 2. Second, sort the sequences by their maximum-dead walk value in decreasing order.

Step 3. Assign the SKUs contained in the highest ranked sequence close to the I/O point.

Step 4. Update the order picking list by eliminating the SKUs which have been already assigned and recalculate the maximum dead-walk.

Step 5. Repeat steps 1-4 until all SKUs have been assigned.

5. Maximum-Dead Walk Algorithm with replenishment. The maximum dead-walk heuristic presented in the previous section can be slightly modified to include replenishment flows. The calculation of the maximum-dead walk for each order picking sequence stays the same, however, it is added the maximum restock walk.

Below are the steps of this heuristic:

Step 1. Sort the SKUs contained in each order sequence in decreasing order of their restock flow.

Step 2. Assign the highest ranked SKU as far as possible from the I/O point and multiple its restock flow by the distance of such storage location from the I/O point (which is basically the number of unassigned Bay locations).

Step 3. Repeat step 2 for the next ranked SKU in the sequence and added to the result obtained in step 2. The summation of all the results becomes the maximum restock walk.

Step 4. Repeat steps 2-3 for all sequences.

Step 5. For each sequence add the maximum restock walk to the maximum dead walk and sort the sequences in decreasing order of their total walk.

Step 6. Assign the SKUs contained in the highest ranked sequence close to the I/O point.

Step 7. Update the order picking list by eliminating the SKUs which have been already assigned, and recalculate the maximum dead-walk and maximum restock walk for each unassigned sequence.

Step 8. Repeat steps 5-7 until all SKUs have been assigned

CHAPTER 4

ANALYSIS OF RESULTS

The results of the BMILP model for different case scenarios are presented in this Chapter. An analysis is described on the differences in order picking and replenishment distances when the assignment of SKUs to locations is both unconstrained and constrained. Further analysis is conducted in order to determine the benefits of integrating replenishment flows in the SKU assignment problem. The BMILP model is also compared to Order Oriented Slotting (OOS) in several case studies considering only order picking distances. Moreover, to validate the quality of the results that can be obtained from the proposed model, the BMILP and Integrated Max-Dead Walk heuristic method are compared to other scientific and popular slotting approaches and it is shown that significant savings in distances can be obtained.

4.1 Unconstrained and Constrained Case Study

In practice, it is important to investigate how a new layout with constraints would perform and how much better it would perform if such constraints were ignored. If the savings for an unconstrained layout are significantly larger compared to a restricted one, it would provide managerial insights that could lead to a relaxation of the layout at the expense of removing such constraints.

Based on the order picking data given in Table 3.1 and the replenishment flow in Table 3.2, the BMILP model is solved for several constrained scenarios and then compared to the unconstrained results. Hereafter, we will refer to Table 3.1 and 3.2 as Exercise 3.1.

4.1.1 Unconstrained Scenario

The solution to Exercise 3.1 using the BMILP model is given in Table 4.1 below and illustrated in Figure 4.1. The value of α used was 0.5, meaning that both order picking and replenishment distances have equal weight.

Table 4.1 Unconstrained solution to Exercise 3.1

Unconstrained	
SKU	Assigned to Bay
2	1
8	2
3	3
1	4
9	5
7	6
4	7
5	8
6	9
Lower Bound	132.5
Solution Found	132.5

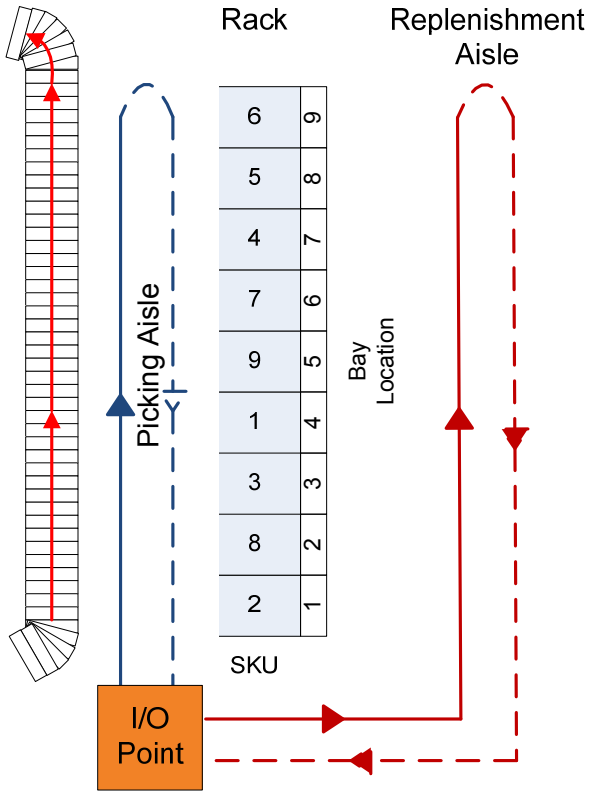


Figure 4.1 Illustration of unconstrained layout solution for Exercise 3.1

The solution found for this problem is the optimal since is equal to the lower bound. However, it should be pointed out that, because of constant α, the solution values are not the actual score of the total distances of order picking and replenishment, so they need to be scored independently. Using the SKU assignment solution of BMILP model, it is possible to score the total distances incurred in order picking and replenishment by calculating: order picking = $\sum_{k=1} L_k F_k$, and replenishment = $\sum_{i=1} D_i R_i$. The specific results for this solution in distances per each activity are shown in Table 4.2 and compared to the original layout. The original layout was based on the simple assignment of SKUs {1,2,3, … ,9} to Bay locations {1,2,3, … ,9}.

Table 4.2 Unconstrained distance results for Exercise 3.1

	Distances			Improvement		
Layout	Order Picking	Restocking	Total	Order Picking	Restocking	Total
Original	165	221	**386**	-	-	-
BMILP Model	156	109	**265**	5%	51%	**31%**

The results above indicate that with the optimized layout order pickers would travel 156 Bays to fulfill the whole set of orders, while restockers would travel 109 Bays to replenish all the SKUs. This represents an improvement of 5% and 51% over the original picking and restocking distances, respectively. Although the savings in picking distance do not appear to be significant, the significant improvements made on the replenishment side helped the overall system to reduce 31% of its original walk, going from 386 Bays to 265 Bays.

4.1.2 Constrained Scenarios

To study the unconstrained and constrained layout behavior, several constrained scenarios were created based on the three additional constraints discussed in Chapter 3: order sequencing constraints, replenishment ranges, and SKU grouping.

Scenario 1: Order Picking Sequencing Constrains.

1. SKU 8 must be picked last a` `t all times because is a soft material.
2. SKU 6 must be picked before SKU 7 to ensure good box space utilization.

Scenario 2: Replenishment Ranges Constrain.

3. SKUs 4, 6, and 9 must be stored within the first five Bays because of storage size requirements.

Scenario 3: Grouping Constrain.

4. SKUs 2, 5, and 9 must be stored next to each other for diverse reasons, i.e., flammable products need to be stored together.

Scenario 4: All of the above.

Formulating the above equations we obtain:

$$D_8 = 9 \tag{1}$$

$$D_6 < D_7 \tag{2}$$

$$1 \leq D_i \leq 5 \quad ; \quad \forall \ i = 4, 6, 9 \tag{3}$$

$$max_G - min_G \leq 2 \quad ; \quad \forall \ G = 1 \tag{4}$$

$$D_i \leq max_G \quad ; \quad \forall \ i = 2, 5, 9 \ \ and \ \ G = 1 \tag{4}$$

$$D_i \geq min_G \quad ; \quad \forall \ i = 2, 5, 9 \ \ and \ \ G = 1 \tag{4}$$

For each scenario, the BMILP formulation was solved with $\alpha = 0.5$. The results of each scenario are shown in Table 4.3.

Analysis of Table 4.3 indicates that at each scenario the assignment of SKUs satisfy the constraints (the constrained SKUs are displayed in gray). As expected, the unconstrained scenario allows more significant savings on the total traveling distances than the constrained scenarios. Also, the improvement potential diminishes as more constraints are added to the new

layout. Nevertheless, the difference in the total distance between the most constrained layout (scenario 4) and the unconstrained one is 10%, with relatively no difference in the restocking distance. In this case, if the costs of removing the constraints in scenario 4 exceeded the 10% savings obtained in walking distances, the implementation of the unconstrained layout would be seriously questioned. For this reason, it is of great value to analyze both constrained and unconstrained scenario results and based upon the difference in savings and the cost of removing the constraints decide which layout to implement.

Table 4.3 Comparison of results between original, unconstrained, and constrained layout scenarios. The SKUs in gray represent those which are constrained.

	Layout	I/O Point	Bay Location 1	2	3	4	5	6	7	8	9	Distances: Order Picking	Restocking	Total	Improve from Original
			SKU Assignment												
	Original		1	2	3	4	5	6	7	8	9	165	221	**386**	
	Unconstrained		2	8	3	1	9	7	4	5	6	156	109	**265**	31%
Constrained	**Scenario 1** (Pick Sequence)		2	3	1	4	9	5	6	7	8	169	113	**282**	27%
Constrained	**Scenario 2** (Restock Range)		2	8	9	4	6	7	5	1	3	175	117	**292**	24%
Constrained	**Scenario 3** (Group)		3	1	2	9	5	7	8	6	4	168	108	**276**	28%
Constrained	**Scenario 4** (All)		2	9	5	6	4	1	3	7	8	195	110	**305**	21%

4.2 BMILP Model Performance Analysis

In this section, the performance of the BMILP model is investigated with respect to other assignment methods in the literature. The BMILP model performance analysis is broken in two parts. First, a comprehensive comparison of the BMILP model with other methods is presented from three different layout optimization perspectives: order picking, replenishment, and both activities. Then, the integrated BMILP model is compared to the integrated Max-Dead Walk heuristic algorithm described in Chapter 3 in terms of quality of solution and solving time.

4.2.1 Optimizing for Order Picking

The following alternative assignment methods are to be compared to the BMILP model with respect to order picking distance:

1) COI, by Heskett (1963);

2) Sequence Popularity;

3) Labor Efficiency, by Bartholdi and Hackman (2007);

4) OOS, by Heragu (2007);

5) Max Delay Algorithm, by Wutthisirisart (2008); and

6) Max Dead-Walk Algorithm.

To analyze the differences in the SKU assignment approach among the alternative methods and the BMILP model, Table 4.4 presents the assignment results of each method for Exercise 3.1 compared to its original layout. For convenience, the order picking sequence Table 3.1 is displayed again here.

Table 3.1 Order Picking Sequences of Exercise 3.1

Sequence	SKU numbers / Bay Number					Frequency
1	1	3				6
2	1	2	4			4
3	5	6	7	8	9	4
4	2	8				4
5	7	9				3
6	4	5	6			3
7	2	6	7	8	9	1
8	1	2	7	8	9	1

Table 4.4 Results of assignment methods for Exercise 3.1

		Bay Location									Distances	
Layout Method	I/O Point	1	2	3	4	5	6	7	8	9	Order Picking	Improvement from total original
		SKU Assignment										
Original		1	2	3	4	5	6	7	8	9	165	
BMILP (only order picking, α=1)		3	1	4	2	8	7	9	6	5	147	11%
COI		1	2	8	7	9	6	4	5	3	176	-7%
Sequence Popularity		1	3	2	8	7	9	6	4	5	154	7%
Labor Efficiency		1	6	8	4	7	2	5	9	3	195	-18%
OOS		9	7	8	6	5	2	4	1	3	171	-4%
Max Delay Algorithm		8	2	1	4	3	7	9	6	5	153	7%

Clearly, the most effective layout comes from the BMILP model and the Max Dead-Walk Algorithm, with 11% less picking walk distance than the original layout. The Sequence Popularity and Max Delay heuristics appear to be the next best methods with 7% improvement

over the original. The good solution and simplicity of the Sequence Popularity heuristic makes it a very good rule of thumb in generating layouts. Nevertheless, the disadvantage of implementing the Sequence Popularity comes when highly frequent sequences contain large number of SKUs, and so its SKUs become candidates to be assigned first at the expense of increasing the lengths of other next-high frequent sequences. The BMILP model evaluates both, the frequency of the sequences and the number of SKUs contained in them.

As expected, some of the alternative assignment methods worsen the picking distances of the original layout due to the nature of their formulations, which fail to capture the sequencing structure of order picking. Per example, COI and Labor Efficiency assign SKU 1 to the closest location to the I/O point, Bay location 1, because of its high demand, while SKU 3 is assigned to the farthest Bay Location from the I/O point because of its relatively low demand compared to the rest of the SKUs. In the end, assigning SKUs 1 and 3 far from each other and far from the I/O point will increase the order picking distance of sequence 1, which is one of the most frequent routes. OOS does a better job on assigning SKUs 1 and 3 next to each other; however, the order picking distance of sequence 1 is increased because both SKUs are assigned at the end of the picking aisle. Contrary to COI, Labor Efficiency, and OOS, the BMILP model seeks to reduce the lengths of order picking sequences; this is why SKUs 1 and 3 were assigned close to each other and near the I/O point, resulting in advantage for order pickers.

To validate the findings found above in the performance of each assignment method, a set of 4 more case problems, each different sequence and bay sizes, were constructed and solved. The case problems were conducted given the assumption of an empty fast pick area. The results of each method are compared to the lower bound (LB), see Table 4.5 and Figure 4.2. In the cases where the BMILP was unable to find the optimal solution, the best feasible solution found

is recorded. Note that the OOS is removed from this part due to its data structure complexity in for large-scale problems, but it is compared later to the BMILP model for a set of smaller size cases.

Observation of Figure 4.2 confirms that for all the case problems the BMILP achieves a better result than the rest of the methods. The BMILP model and the Max Delay Algorithm achieve steady results from the LB, oscillating within 20% of the LB. The most unstable method is the Max-Dead Walk algorithm which in the first 4 cases performed within 20% of the LB, but in case 5 jumped to almost 40%. Not much surprisingly, the COI, Labor Efficiency, and Sequence Popularity methods reported the highest results.

For further analysis, the case problems were intentionally constructed with two different classes of order picking sequence frequency distributions. The first class, cases 2 and 4, contains frequencies which are not widely distributed, meaning that they are pretty much constant throughout the sequence list. The second class, cases 1, 2, and 3, contains highly distributed frequencies. It can be observed that for the first class of cases, COI, Labor Efficiency, and Sequence Popularity obtain decent approximations to the LB, but as the frequencies become more spread in the order picking process it is very likely that these methods will lead to higher deviations from the LB.

Table 4.5 Comparison of assignment methods to the lower bound for each of the 5 case problems.

Case	Sequences	Bays	Lower Bound	BMILP Model	COI	Sequence Popularity	Labor Efficiency	Max Delay	Max Dead-Walk
1	8	9	147	147	176	154	195	153	147
2	20	26	1409	1415	1516	1503	1552	1503	1480
3	30	44	11455	12059	14805	14521	15455	12520	13442
4	60	65	4011	4061	4408	4387	4638	4299	4064
5	50	91	43265	51069	65978	72098	68441	51114	59336

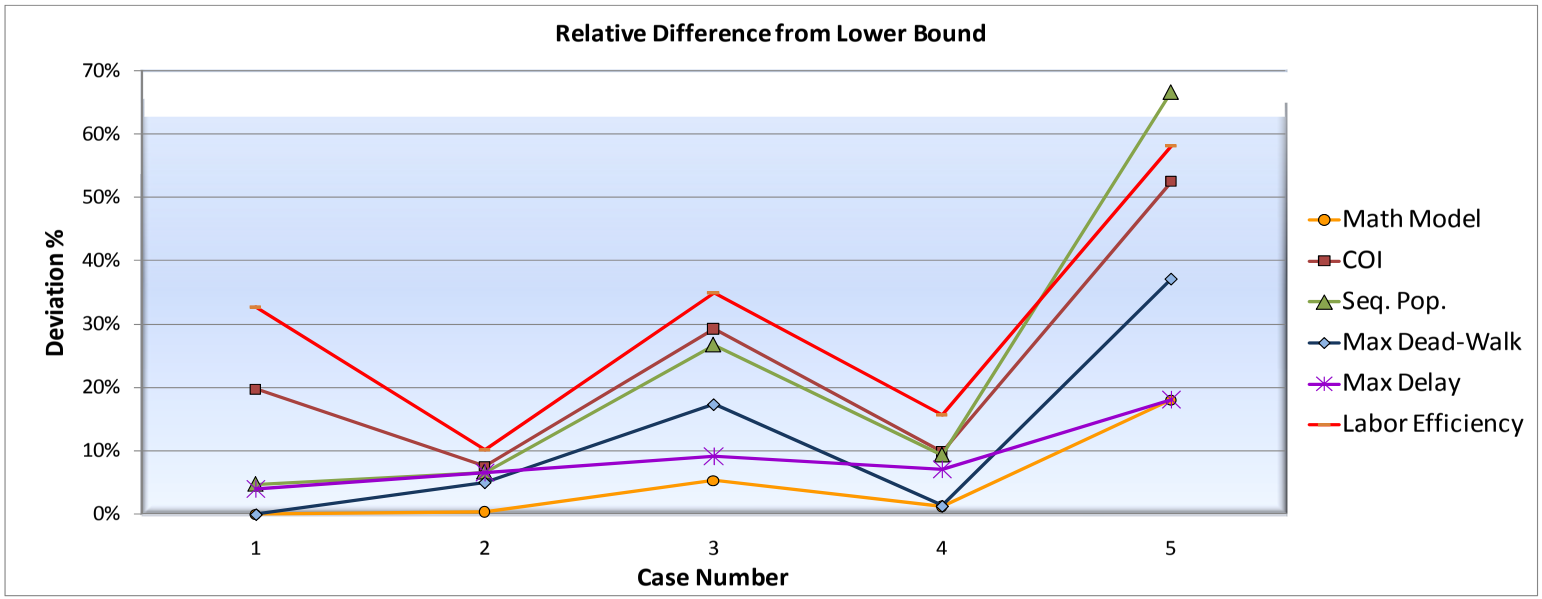


Figure 4.2 Relative difference from the Lower Bound for the various case problems

4.2.1.1 BMILP model versus Order Orienting Slotting

Considering that OOS is the most order picking sequence oriented model among the rest of the alternative assignment methods, the BMILP model is compared to it. The purpose of this comparison is to validate the ability of the BMILP model itself to find the optimal solution that globally minimizes the order picking distances in a fast pick area. Thus, 6 short problems of less than 10 SKUs were constructed to guarantee optimal solutions for both methods. Their optimal solutions are compared in Table 4.6.

Table 4.6 Comparison of optimal solutions between OOS and the BMILP model.

	Problem Size		Picking Distance		
Case	**Sequences**	**Bays**	**OOS**	**Model**	**% Difference**
1	8	9	171	147	14%
2	10	10	287	267	7%
3	10	10	358	318	11%
4	10	10	413	406	2%
5	10	10	419	406	3%
6	10	10	324	305	6%
				Average	**7%**

Per the table above, in all instances the BMILP obtained a better solution than the OOS; an average of 7% improvement from the OOS solution. These results indicate that the formulation of the BMILP model assigns SKUs in the fast picking area in a better way than OOS. Therefore, it is concluded that the BMILP model is more representative of the problem structure in fast picking areas than that approximated by OOS.

4.2.2 Integration Analysis

As we have learned, previous research has mostly focused in reducing costs from a myopic perspective, either analyzing the flows from order picking or restocking. Here, the importance of integrating these two different, but, very connected activities is demonstrated, and it is shown that additional savings can be obtained from such integration.

In section 4.2.1, Exercise 3.1 was solved with respect to order picking flows ignoring the replenishment flow that occurs in the fast picking area. Such an approach attempts to benefit the order picking distances; however, it is unknown how this approach impacts the replenishment distances and, therefore, the total distances of the fast pick area if both were considered. In order to answer this question, Table 4.7 presents the results previously reported in Table 4.4 with the inclusion of the scores of the replenishment distances for each assignment method. In addition, the integrated methods: integrated BMILP model and integrated Max Dead Walk algorithm are evaluated for this exercise as well. Also, the results of the BMILP model considering only restocking is presented.

The integrated BMILP model was solved to optimality for Exercise 3.1. Clearly, it can be seen that the results for the overall distances obtained in the Integrated BMILP outperforms the rest of the models. This is achieved because there is a substantial improvement in the replenishment distances which the other methods were unable to capture, except for the BMILP model based on restocking ($\alpha = 0$), which, as expected, significantly reduced the restocking distances at the cost of substantially increasing the picking distances.

Table 4.7 Update of Table 4.4 after evaluating replenishment flows for each assignment method based on order picking for Exercise 3.1.

	Layout Method	I/O Point	Bay Location 1	2	3	4	5	6	7	8	9	Distances: Order Picking		Restocking		Total	Improvement from total original
			SKU Assignment														
	Original		1	2	3	4	5	6	7	8	9	165	+	221	=	**386**	
Integrated	**BMILP Integrated (α=0.5)**		2	8	3	1	9	7	4	5	6	156	+	109	=	265	31%
	Max Dead-Walk Integrated		2	9	7	8	6	3	1	5	4	174	+	102	=	276	28%
	BMILP (only order picking, α=1)		3	1	4	2	8	7	9	6	5	147	+	124	=	271	30%
	BMILP (only restock, α=0)		3	1	4	2	8	7	9	6	5	182	+	91	=	273	29%
	COI		1	2	8	7	9	6	4	5	3	176	+	129	=	305	21%
	Sequence Popularity		1	3	2	8	7	9	6	4	5	154	+	122	=	276	28%
	Labor Efficiency		1	6	8	4	7	2	5	9	3	195	+	161	=	356	8%
	OOS		9	7	8	6	5	2	4	1	3	171	+	123	=	294	24%
	Max Delay Algorithm		8	2	1	4	3	7	9	6	5	153	+	129	=	282	27%

4.2.2.1 Integration Benefits

From Table 4.7, it can be concluded that additional savings can be obtained in the overall distances of the fast pick area if order picking and replenishment flows are taken into consideration for the SKU assignment problem. However, it remains unknown if this statement will be true for all cases. For this reason, an integration sensitivity analysis is performed for the same 5 cases presented in Table 4.5. For each case, the BMILP is solved for $\alpha = 1$, when replenishment is not integrated; and for $\alpha = 0.5$ when both activities are integrated. The results are presented in Table 4.8.

The Integrated BMILP model was only able to find the optimal solution of cases 1 and 2, for the rest of the cases the best feasible solution was recorded along with the lower bound found. According to Table 4.8, the integrated approach generates an average of 4% additional savings in the overall distances incurred in the fast picking area for all cases. In case 3 the Integrated BMILP model presents a -0.2% deficit compared to the non-integrated approach. Nevertheless, for the cases in which the Integrated BMILP model did not find the optimal solution, the lower bounds found shows that there could be potential room for improvement.

Moreover, the potential additional savings that could be obtained by the Integrated BMILP depend upon the nature of the fast picking area under study. If the replenishment flow is close or higher than the order picking flow, it would be expected that more significant savings would be obtained using the integrated approach. Per example, in case 2 and 4, the replenishment walk distance is higher than the order picking walk distance for the non-integrated BMILP model and, therefore, the overall walking distance figures an improvement when solved in the integrated approach.

Table 4.8 Comprehensive comparison of distance results between the Non-Integrated BMILP and the Integrated BMILP

	Problem Size		Non-integrated BMILP α=1			Integrated BMILP α=0.5				Integrated BMILP α=0.5 Lower Bound	
Case	Sequences	Bays	Order Picking	Replenishment	Total	Order Picking	Replenishment	Total	Total % from α=1	Total	Total % from α=1
1	8	9	147	124	271	156	109	265	2%	265	2%
2	20	26	1415	1766	3181	1661	1264	2925	8%	2925	8%
3	30	44	12059	5766	17825	12306	5560	17866	-0.2%	17147	4%
4	60	65	4061	10453	14514	4941	8242	13183	9%	13169	9%
5	50	91	51069	23880	74949	50701	22863	73564	2%	66561	11%
								Average	**4.2%**		

4.2.2.2 Integrated BMILP model versus Integrated Max Dead-Walk Heuristic

As it was discussed in the previous section, the Integrated BMILP model was only able to find a feasible solution for large problems over 26 SKUs. Due to the discrete integer linear formulation of the BMILP model, it becomes evident that for larger problems one has to resort to heuristics to approximate the solution to the LB in reasonable computation time.

In this section, the Integrated BMILP model and the Integrated Max-Dead Walk Heuristic are compared with respect to their computation time and the quality of the solutions obtained with respect to the LB. Table 4.9 summarizes the comparison of results for each case problem.

As expected, the solution time for the heuristic is significantly less compared to the Integrated BMILP model; it required less than a second to solve each case. This gives the heuristic the advantage of being able to solve real case problems with thousands of SKUs. However, the quality of the solution obtained by the heuristic deviates farther from the LB as the problem size increases, reducing the accuracy of the SKU assignment to properly minimize the picking and restocking distances. The need for faster and more accurate heuristics is further discussed in Chapter 5.

Table 4.9 Comparison of deviation from LB and computation time for Integrated BMILP and Integrated Max Dead-Walk Heuristic

	Problem Size		Integrated BMILP α=0.5						Integrated Max Dead-Walk Heuristic				
Case	Sequences	Bays	Total Lower Bound	Order Picking	Replenishment	Total	% from LB	Run Time (seconds)	Order Picking	Replenishment	Total	% from LB	Run Time (seconds)
1	8	9	265	156	109	265	0.0%	2	174	102	276	4.2%	0.003
2	20	26	2925	1661	1264	2925	0.0%	97	1776	1405	3181	8.8%	0.022
3	30	44	17154	12505	5620	18125	5.7%	185	15622	5494	21116	23.1%	0.036
4	60	65	13169	4941	8242	13183	0.1%	332	5862	9153	15015	14.0%	0.048
5	50	91	66561	50701	22863	73564	10.5%	656	79316	22258	101574	52.6%	0.067

4.3 Sensitivity Analysis

The problem structure presents three important factors:

A. Number of SKUs
B. Number of sequences
C. Frequency distribution of orders

In this section, a sensitivity analysis is performed on these factors, A, B and C, to determine in which way they are significant to the BMILP model computation time. For the sensitivity run, the factors were combined into high and low levels as follows:

- High A, High B, High C
- High A, High B, Low C
- Low A, High B, High C
- Low A, High B, Low C

The result of the sensitivity run with the above combination of factors is displayed in Figure 4.3. In the Figure, the x-axis indicates the problem size solved; it represents number of SKUs when solved for High A combinations, or number of Sequences when solved for Low A combinations. In the case where the problem was not solved to optimality in a reasonable amount of time or because the solution did not improve after 5 minutes, the best solution was recorded along with its deviation percent from the LB found.

According to the results it can be observed that the combinations with Low A do not appear to be significant in the computation time. Thus, it is relatively fast to solve the SKU assignment problem when the problem structure contains very few SKUs. However, when the

level of A is high there is a significant increase in the computation time. And if combined with high levels of C, the computation time increases even more than with low levels of C.

From the sensitivity of analysis it can be concluded that the problem structure becomes complex when all of the described factors above are at high levels, meaning large number of SKUs and order sequences, and a disperse distribution in the frequency of orders. In such case, it is expected that the computation time will present an exponential growth.

When solving an SKU assignment problem with large number of SKUs (over 100), a relaxation to the problem would be to create groups or zones of SKUs in order to reduce the number of variables and achieve approximate solutions in less computation time.

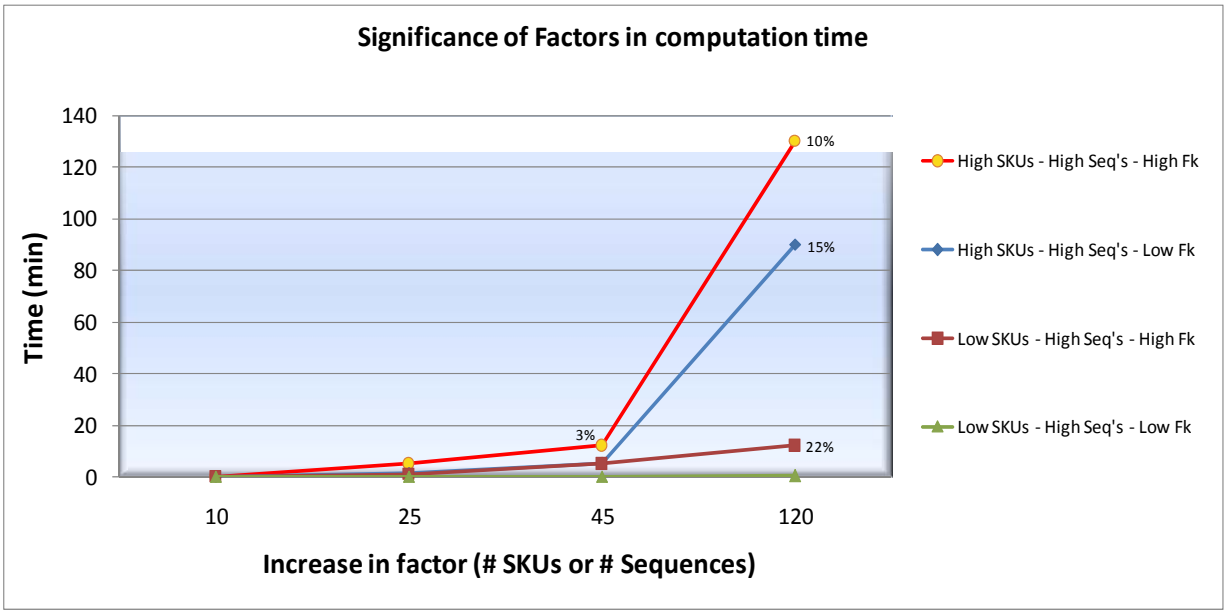


Figure 4.3 Significance of factors in the BMILP model's computation time.

CHAPTER 5

CONLUSIONS AND FUTURE RESEARCH

5.1 Research Summary

Order picking and replenishment in a fast picking area are the most labor-intensive and costly activities of any DC, as traveling accounts for up to 50% of the total labor time. Therefore, this research focused on SKU slotting planning, also known as the assignment problem, to optimize travel distances. The previous SKU assignment literature has primarily focused on the assignment of SKUs from an order picking perspective. A critical observation of previous assignment methods such as COI, Labor Efficiency, QAP, and Correlated Assignment reveals that they fail to capture the sequencing structure of order pickers in ordinary fast picking operations. The most relevant assignment strategy with respect to order sequencing is OOS, however, it was found that it does not work well when SKUs are picked in different orders is common practice. In addition, replenishment has not been integrated in the SKU assignment problem; instead, it has been formulated separately to minimize the number of restocks and the traveling involved from bulk warehouse to the fast pick area.

Therefore, based on the observation that the SKU assignment for items in manual sequenced order-picking is a relatively new warehousing application, and that replenishment has not yet been considered in the slotting planning, this research developed a SKU assignment

model that is able to minimize the walking distances of order pickers and restockers in a fast picking area by capturing the sequencing factors of the order picking process.

In this research, the integrated SKU assignment problem was formulated as a mathematical Binary Mixed Integer Linear Programming (BMILP) model for the S-shaped routing policy. The BMILP model was successfully solved using the LINGO solver. It was observed that the BMILP model runs in non-polynomial time.

To analyze the performance of the BMILP model, it was tested against previous assignment models and several heuristics on a series of case problems. The performance analysis was broken in two parts: first, a comprehensive comparison based on optimization for order picking; second, a comparison of the differences between an integrated and a non-integrated BMILP model. In the first part, the BMILP model outperformed the rest of the methods and achieved results within 20% of the lower bound (LB). For the second part, the integrated BMILP achieved, on average, better results than the non-integrated BMILP. To validate the optimality of the BMILP formulation, the BMILP model was compared to the OOS for several small cases.

Finally, the Max Dead Walk Algorithm was tested against the Integrated BMILP model to compare the differences in the computation time and the quality of the solution. It was observed that the Max Dead Walk heuristic required less computation time, but the quality of the solutions deviated from the LB as the problem size increased.

5.2 Conclusions

According to the run time analysis conducted on the BMILP model, it is concluded that the model runs in non-polynomial time, since the computation time increases exponentially as the problem size increases. The BMILP model was able to solve to optimality small size problems with less than 30 SKUs, for problem sizes of less than 150 SKUs it could only obtain a feasible solution after 5 minutes of no improvement in the solution, and for problems higher than 150 SKUs the model could not reach a feasible solution in a reasonable amount of time. The limits of the Integrated BMILP model become evident in this regard, thus, the need to develop fast and accurate heuristics is of great importance in order to solve large realistic size problems that involve thousands of SKUs and hundreds of different sequences.

From the series of case problems analyzed, it was evident that the integrated BMILP model assigns SKUs in a more accurate way such that the distances involved in order picking and restocking are minimized. The solutions obtained by the BMILP model delivered less walking distance compared to the rest of the methods tested. This is partially obtained because the formulation of the BMILP model attempts to reduce the route lengths of each order, which make more sense to order pickers; as opposed to other practical methods that formulate the assignment either on a SKU turnover or correlation basis without looking at the order content.

The validation of the optimality of the BMILP model was confirmed after comparing it to OOS. The solutions of the BMILP model reached lesser values than the OOS. This means that the solution approach formulated by the BMILP model is more appropriate for the structure we are addressing.

The integration analysis proved the importance of integrating order picking and replenishment in the SKU assignment problem. From the series of case problems, the Integrated BMILP model obtained additional savings in the overall distance, 4% on average. This is achieved because the Integrated Model substantially reduces the restocking distances involved which a non-integrated approach would ignore. Moreover, it is concluded that the potential additional savings that could be obtained by the Integrated BMILP model vary depending on the replenishment flow of the problem. If the replenishment flow is low, it is apparent that an integration approach would not substantially improve the overall distance; but if the replenishment flow is high, more significant savings in the overall distances are likely to be obtained.

In terms of heuristics, the Integrated Max Dead Walk Algorithm experiences results close to the LB for small size problems, but increasingly deviates from it when the problem size increases. For problems with just order picking, the Max Delay Algorithm appears to be a promising tool for the SKU assignment problem. Its solutions constantly fell within 20% of the LB for each of the problem sizes tested, which makes it a tool worth testing in larger environments.

5.3 Future Research

In the future, some of the important aspects of multi-aisles fast picking areas that should be included within the BMILP model are:

1. The ability to capture return distances to the I/O point after an order is fulfilled
2. Consideration of separate replenishment I/O points per replenishment aisles

With respect to point a) it can be seen that as a picker enters a picking aisle, where the S-shaped route flow goes in the direction to the I/O point, the SKU at the end of the aisle is actually the closest to the I/O point at a given return. In this regard, it would be interesting to see the impact of penalizing the assignment of SKUs far from the I/O point. With respect to b), the modeling of automated replenishment systems, i.e. divert conveyors, could be achieved.

Another aspect to explore in the SKU assignment problem is the case of dynamic relayout with reprofiles. As it is well known in DCs, the demands of SKUs fluctuates over time, so a given layout might not be accurate for later times when orders tend to visit other SKUs more frequently than they did before. The extension of the BMILP model to a dynamic relayout could then study the cost-benefit of reprofiliing versus keeping the same layout. In this same direction, considering that relayouts in a fast pick area are costly and very time consuming, the development of a SKU assignment model that optimizes walking distances with minimum SKU movement would definitely be a very practical tool for existing DCs.

REFERENCES

[1] Council of Supply Chain Management Professionals, 19th Annual State of Logistics Report.

Alagoz, O., Norman, B.A, and Smith, A.E. (2008) Determining aisle structures for facility designs using a hierarchy of algorithms. *IIE Transactions*, 40, 1019-1031.

Bartholdi, J.J. and Hackman, S.T. (2008) *Warehouse and Distribution Science*, release 0.87. Not yet published.

Bartholdi, J.J., and Platzman, L.K. (1986) Retrieval strategies for a carousel conveyor. *IIE Transactions*, 18, 166-173.

Bozer, Y.A. (1985) Optimizing throughput performance in designing order picking systems, Ph.D. Dissertation, Georgia Institute of Technology, Atlanta, GA.

Bozer Y.A., Meller, R.D., and Erlebacher, S.J. (1994) An improvement-type layout algorithm for single and multiple-floor facilities. *Management Science*, 40(7), 918-932.

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

Elsayed, E.A, Lee, M.K., Kim, S., and Scherer, E. (1993) Sequencing and batching procedures for minimizing earliness and tardiness penalty of order retrievals. *International Journal of Production Research*, 31(3), 727-738.

Frazelle, E.H., Hackman, S.T., Passy, U., and Platzman, L.K. (1994) The forward-reserve problem. *Optimization in Industry 2*, Ciriani, T.A and Leachman R.C. (Eds.), John Wiley & Sons, 43-61.

Frazelle, E.H., and Sharp, G.P. (1989) Correlated assignment strategy can improve order picking operation, *Industrial Engineering*, 4, 33-37.

Frazelle, E.H. (2002) *World-class warehousing and material handling*. McGraw-Hill, New York, NY.

Goldratt, E.M. (2004) *The Goal*, 3rd edition, North River Press, Great Barrington, MA.

Hackman, S.T., and Platzman, L.K. (1990) Near optimal solution of generalized resource allocation problems with large capacities. *Operations Research*, 38(5), 902-910.

Hackman, S.T., and Rosenblatt, M.J. (1990) Allocating items to an automated storage and retrieval system. *IIE Transactions*, 22(1), 7-14.

Hausman, W.H., Schwarz, L.B., and Graves, S.C. (1976) Optimal storage assignment in automatic warehousing systems. *Management Science*, 22(6), 629-638.

Heskett, J.L. (1963) Cube-per-order index – a key to warehouse stock location. *Transportation and Distribution Management*, 3, 27-31.

Heskett, J.L. (1964) Putting the cube-per-order index to work in warehouse layout. *Transportation and Distribution Management*, 3, 23-30.

Hollingsworth, B.K. (2003) Decision strategy to minimize replenishment costs in a distribution center with forward-reserve storage. M.S. Thesis, Ohio University, OH.

Hwang, H., and Song, J.Y. (1993) Sequencing picking operations and travel time model for man-on-board storage and retrieval warehousing system. *International Journal of Production Economics*, 29, 75-88.

Jernigan, S.A. (2004) Multi-tier inventory systems with space constraints. Ph.D. Dissertation, Georgia Institute of Technology, Atlanta, GA.

Kallina, C., and Lynn, J. (1976) Application of the cube-per-order index rule for stock location in a distribution warehouse. *Interfaces*, 7(1), 37-46.

Kim, B., Heragu, S.S., Graves, R.J., and Onge, A.ST. (2003) Realization of a short cycle time in warehouse replenishment and order picking. *International Journal of Production Research*, 41(2), 349-364.

Koopmans, T.C, and Beckmann, M.J. (1957) Assignment problems and the location of economic activities. *Econometrica*, 25, 53-76.

Le-Duc, T., and De Koster, R. (2005) Layout optimization for class-based storage strategy warehouses. *Supply Chain Management – European Perspectives*, de Koster, R., Delfmann, W. (Eds.), CBS Press, Copenhagen, 191-214.

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

Montreuil, B., Ratliff, H.D., and Goetschalckx, M. (1990) A modeling framework for integrating layout design and flow network design. *Proceedings of the Material Handling Res. Colloquium*, Hebron, KY, 43-58.

Morse, P.M. (1972) Optimal linear ordering of information items. *Operations Research*, 20(4), 741-751.

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

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

Saab, Y., and Chen, C., (1994) An effective solution to the linear placement problem. *VLSI Design*, 2(2), 117-129.

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

Van den Berg, J.P., Sharp, G.P., Gademann, A.J.R.M, and Pochet, Y. (1998) Forward-reserve allocation in a warehouse with unit-load replenishments. *European Journal of Operational Research*, 111, 98-113.

Wutthisirisart, P. (2008) Linear Sequencing Algorithms. Working paper, University of Missouri-CELDi, Columbia, MO.

APPENDIX

A1. Tables of Case Studies for Performance Analysis

Table A.1 Problem structure for 20 Sequences – 26 SKUs.

Sequence	SKUs				Frequency
1	20	56			303
2	48	49	53		14
3	21	40			6
4	20	22	56		5
5	18	30			4
6	20	21	56		4
7	20	21	22	56	4
8	3	11	19	38	4
9	2	35			3
10	4	36			3
11	23	58			3
12	20	56	58		3
13	20	51	56		3
14	20	56	61		3
15	30	39			2
16	30	38			2
17	10	35			2
18	2	30			2
19	31	33			2
20	3	39			2

Table A.2 Problem structure for 30 Sequences – 44 SKUs.

Sequence	SKUs							Frequency
1	13	40						20
2	35	36	39					17
3	14	33						15
4	13	15	40					30
5	11	23						12
6	13	14	40					25
7	13	14	15	40				33
8	2	6	12	31				40
9	1	28						5
10	3	29						10
11	16	41						15
12	13	40	41					8
13	13	38	40					26
14	13	40	43					5
15	23	32						23
16	23	31						9
17	5	28						12
18	1	23						36
19	24	26						21
20	2	32						12
21	35	37	39					24
22	10	13	40					19
23	17	21	22	24				35
24	13	14	16	34				48
25	2	27	30	32				23
26	14	15	19	40				14
27	11	20	25	30				27
28	14	15	19	40	42			45
29	4	7	8	9	18	32		50
30	13	40	44					13

Table A.3 Problem structure for 60 Sequences – 65 SKUs.

Sequence	SKUs						Frequency
1	20	56					303
2	48	49	53				14
3	21	40					6
4	20	22	56				5
5	18	30					4
6	20	21	56				4
7	20	21	22	56			4
8	3	11	19	38			4
9	2	35					3
10	4	36					3
11	23	58					3
12	20	56	58				3
13	20	51	56				3
14	20	56	62				3
15	30	39					2
16	30	38					2
17	10	35					2
18	2	30					2
19	31	33					2
20	3	39					2
21	22	44					2
22	29	39					2
23	6	17					2
24	20	65					2
25	32	39					2
26	20	57					2
27	41	51					2
28	21	48					2
29	8	57					2
30	1	39					2
31	29	31					2
32	20	56	64				2
33	20	56	63				2
34	15	20	56				2
35	20	41	56				2
36	20	46	56				2
37	44	45	54				2
38	43	47	52				2
39	21	22	41				2
40	7	20	56				2
41	20	56	59				2
42	9	20	56				2
43	21	23	55				2
44	48	50	53				2
45	16	20	56				2
46	24	28	29	31			2
47	20	21	23	44			2
48	3	34	37	39			2
49	21	22	26	56			2
50	18	27	32	37			2
51	21	22	26	56	60		2
52	5	12	13	14	25	39	2
53	42	65					1
54	22	65					1
55	56	61					1
56	3	27					1
57	3	38					1
58	2	39					1
59	22	59					1
60	22	52					1

Table A.3 Problem structure for 50 Sequences – 91 SKUs.

Sequence	SKUs																					Frequency
1	7	33																				30
2	36	37	39	68																		20
3	36	89																				17
4	5	24	25	27	45	62																15
5	59	62																				30
6	36	83																				12
7	65	74																				25
8	2	60	61	62																		33
9	37	38	47	81																		40
10	67	69	75																			5
11	37	70																				10
12	37	38	65																			15
13	9	36	81																			8
14	34	52	59	61																		26
15	36	81	85																			5
16	17	36	81																			23
17	11	83																				9
18	37	39	80																			12
19	70	72	76																			36
20	1	62																				21
21	32	36	81																			12
22	53	58																				24
23	36	80	89	90																		19
24	12	66	82																			35
25	11	40	63	81	82																	48
26	14	17	71	79	80																	23
27	66	89																				14
28	4	15	19	36	37	38	80															27
29	36	37	38	40	80	81	82	89														45
30	36	37	38	66	77	79	80	81														50
31	38	89																				13
32	37	38	74	82	85	87	89	90														53
33	36	37	40	79	80																	86
34	4	17	66	77	78	87																14
35	20	77	78	85	89																	69
36	4	10	11	13	17	18	37	38	40	79	80	82	89									70
37	4	6	16	36	38	39																10
38	36	40	66	80	82																	8
39	17	36	66	73	80	82																83
40	4	6	7	9	18	32	36	37	70	80	83	84	85	89								6
41	1	2	35																			70
42	36	37	38	66	77	78	81	87	89													72
43	4	17	18	20	77	78	80	81	82	86	88	90										10
44	17	36	77	79	89																	94
45	21	22	23	82																		8
46	4	11	20	38	71	85																59
47	3	8	26	28	29	30	31	34	41	42	43	44	46	48	49	50	51	54	55	56	57	87
48	4	11	17	20	37	40																21
49	40	64	79	81																		29
50	37	77	85	86	91																	34

Table A.4 Case 1 for BMILP model vs OOS.

Sequence	SKUs					Frequency
1	1	3				2
2	4	5	6			8
3	1	4				6
4	7	9	10			3
5	1	5				3
6	5	6	8			7
7	2	4	6	8		9
8	1	3	5	7		4
9	1	10				5
10	3	6				1

Table A.5 Case 2 for BMILP model vs OOS.

Sequence	SKUs					Frequency
1	7	10				6
2	4	8	9			5
3	1	5	10			8
4	2	6	8	9		6
5	4	7				5
6	3	9				1
7	7	9				3
8	1	2	3	4		9
9	2	6	7			3
10	2	3				3

Table A.6 Case 3 for BMILP model vs OOS.

Sequence	SKUs							Frequency
1	5	7	9					6
2	1	3	4					1
3	1	2	4	5				8
4	3	4	6	7				3
5	2	3	7	9				5
6	4	6	9	10				5
7	2	3	7	8				9
8	4	5	6	9	10			8
9	3	4	5	7	8	10		5
10	1	4	8					4

Table A.7 Case 4 for BMILP model vs OOS.

Sequence	SKUs							Frequency
1	1	5						3
2	5	6	8					7
3	2	4	6	8				9
4	1	3	5	7				4
5	1	10						5
6	3	6						1
7	4	6	9	10				5
8	2	3	7	8				9
9	4	5	6	9	10			8
10	3	4	5	7	8	10		5

Table A.8 Case 4 for BMILP model vs OOS.

Sequence	SKUs						Frequency
1	3	9					1
2	7	10					3
3	1	2					6
4	5	6					3
5	2	3					3
6	5	7	8				6
7	1	3	4				2
8	1	2	4	5			8
9	3	4	6	7			5
10	2	3	7	9	10		10