Dynamic Slotting and Cartonization Problem in Zone-based Carton Picking Systems

by

Byung Soo Kim

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

Auburn, Alabama
December 18, 2009

Keywords: Warehouse control, Slotting, Cartonization, Zoning,
Order picking system, Dynamic warehouse replenishment

Approved by

Jeffrey S. Smith, Chair, Professor of Industrial & Systems Engineering
Robert L. Bulfin, Professor of Industrial & Systems Engineering
Kevin R. Gue, Associate Professor of Industrial & Systems Engineering

Abstract

Due to the popularity of internet ordering and intelligent logistic and supply chain management systems, customers tend to order more frequently, in smaller quantities, and they require more customized service. As a result, the turn-over rate of SKUs in many warehouses is significantly increasing. The distribution center in this study is zone-based carton picking system and it is dynamically replenished with specific SKUs for next pick wave after pickers complete the picking for the current pick wave. In other words, the picking area is completely reslotted between each pick wave. In this distribution center environment, the long-term demand is of limited value in determining the appropriate assignment of SKUs to slots and items to cartons for the specific pick wave. Thus, the distribution center has two NP-hard assignment problems: slotting –assigning SKUs to slots in the picking area; and cartonization – assigning individual items to cartons. The two primary assignment problems are interrelated and are simultaneously solved at the beginning of the pick wave.

The primary objective in this dissertation is to develop an efficient iterative heuristic methodology for systematically solving two interrelating complex decision problems based on simulated annealing slotting heuristic using correlated SKUs and cartonization heuristic using bin-packing heuristic considering slotting. The proposed heuristic improves the performance of makespan of pickers assigned in each zone compared to two independent heuristics being given does not guarantee a good solution.

Acknowledgments

Earning Ph.D degree was the most challenging work for me. It seems that I have to travel an endless journey. Finally, I am in a happy-ending moment that I successfully complete the journey. It is the right time to appreciate about the people who have contributed to my dissertation.

First of all I would like express the deepest gratitude to my farther. Now, he is very old (83 years old). This dissertation is expected to be the most valuable and the last gift that is presented to him. He had endless supports to me during the M.S. and Ph.D graduate study periods. Without his support and encouragement, I could not achieve an honorable moment. I also would like express thanks to my wife, Eun Jung, my daughter, Bo kyung, and my son, Hyun-oh. They have been a tremendous source of encouragement during my up-and-down Ph.D graduate study period. They always trusted me and supported me without complaining during the period.

In academic area, I would like express my sincere gratitude to my advisor, Dr. Jeffrey S. Smith. He directed me numerous attitudes as an academic researchers. He also provided me invaluable academic suggestions and insights to break through the problems that I met during the dissertation. Sometimes, he showed patients for me that I could completely digest and follow the insight of problems. Sometime, he trimmed useless branches of my dissertation and escorted me in direct/indirect ways to generate productive branches for my dissertation. It was very stressful and time-consuming process. But I realized I became a real researcher after I overcame the threshold in step by step. Without his guidance, encouragement, and patience, I would not meet this successful moment.

I am also indebted to the other committee members, Dr. Robert. L. Bulfin and Dr. Kevin R. Gue. I also express thanks to Dr. Chan S. Park. He suggested me appropriate comments to resolve whenever I was in trouble during degree time. I also express special acknowledgement to Dr. Alice E. Smith, Chair of Department of Industrial & Systems Engineering in Auburn University to help to successfully complete my study.

Finally, I express my sincere gratitude to Dr. Shie-Gheun Koh, Professor of Department of Systems Management & Engineering in Pukyong National University. He always provided me invaluable academic supports and suggestions. Without his academic guidance and personal encouragement, I would not even extend to study an academic graduate program in the United State.

Table of Contents

List of Tables

List of Figures

Chapter 1
Introduction

Inventory, which exists because of a mismatch between supply and demand, is an important supply chain driver because changing inventory policies can dramatically alter the supply chain's efficiency and responsiveness. Therefore, the warehouse that stores inventory also plays an important role in supply chain management. According to the 19th Annual State of Logistic Report sponsored by the Council of Supply Chain Management Professionals, U.S business logistics costs hit $1.4 trillion in 2007. In addition, warehouse-related costs, which make up 9.9% of the total logistics cost, are approximately $100 billion (Council of supply chain management professionals, 2008). This means that managing the warehouse efficiently is essential to reduce logistics cost in a supply chain.

Frazelle (2002) classified the warehouse into seven types, which includes raw material warehouse, work in process warehouse, finished goods warehouse, distribution warehouse (or distribution center), fulfillment warehouse, local warehouse, and value-added service warehouse. The first three types store raw materials, work in process, and finished goods, respectively. The distribution warehouse accumulates and consolidates products from various points of manufacture within a single firm, or from several firms, for combined shipment to common customer. The goal of a fulfillment warehouse is to receive, pick, and ship small orders for individual consumers. The local warehouses are distributed in the field in order to shorten transportation distances and permit rapid response to customer demand. In the value-added service warehouse, finally, some product customization activities are executed, including packaging, labeling, pricing, and returning processes. Among these types, we focus mainly on the distribution warehouse (See Figure 1.1 for a description).

The decision problems for the distribution warehouse can generally be classified into three categories according to the timeframe of decisions needed. The first decision problem is to find the location(s) of warehouse(s). If a firm is designing the logistics network, it has to decide the number of distribution warehouses and their locations to minimize the service time for customers and/or to minimize transportation cost. Once the location of a distribution warehouse is found, the next decision problem to be solved is designing the warehouse configurations. This problem consists of two main research areas: overall warehouse design and internal warehouse design. In the overall warehouse design area, the physical warehousing system is constructed by selecting appropriate storage facilities (e.g. block stacking, single-deep lane storage, double-deep lane storage, carton flow-rack, AS/RS, carton/case picking, small item picking, etc.) and material handling equipment (e.g. fork-lift, conveyor, hoist, stacker crane, etc.), while the internal layout of the warehouse is made through solving the internal warehouse design problems.

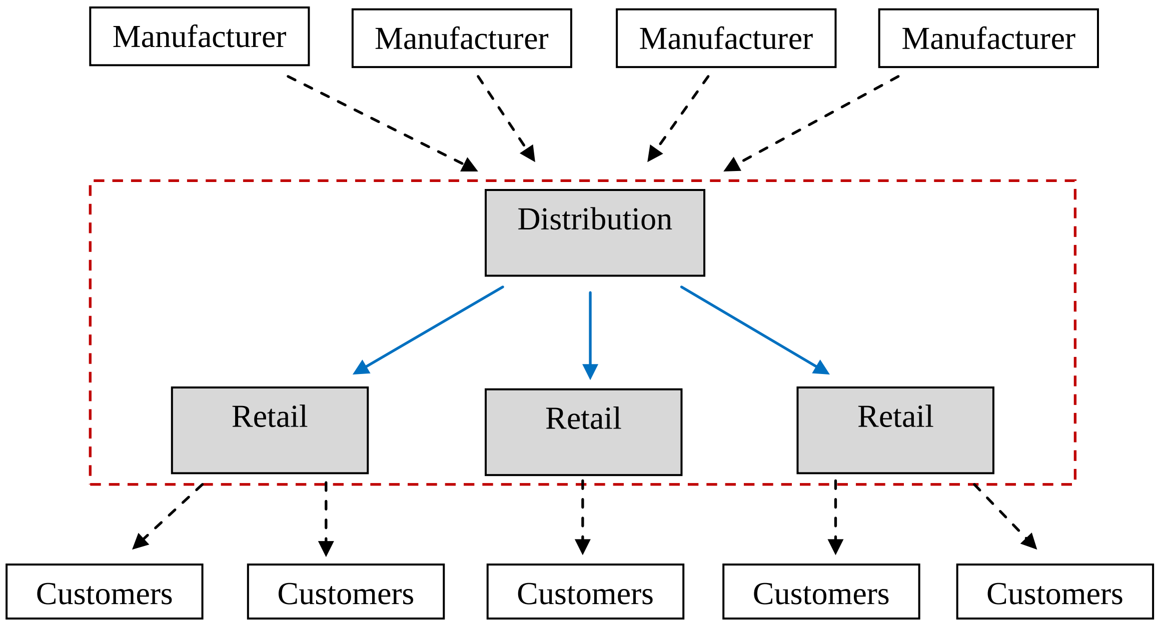


Figure 1.1 Description of distribution warehouse in supply chain

The above two decision problems are long-term strategic and/or mid-term tactical problems in a supply chain management. However, the third decision problem includes short-term operational problems, which are the main focus of this dissertation. The warehouse managers are most frequently faced with this kind of problems. The warehouse operation problem can be classified into four areas according to four main operations of a warehouse as follows:

1) Receiving: This operation is the collection of activities involved in the receipt of all products coming into the warehouse.

2) Storing: This operation is assigning storage space to inventory items. Three

fundamental decisions are introduced for storing operation such as how much inventory should be kept for a SKU (stock keeping unit) in the warehouse, how frequently or when should the inventory for a SKU be replenished, and where should the individual SKUs be stored in the warehouse. The first two decisions belong to the traditional inventory control area. Throughout this dissertation, we confine the storing operation to third decision of the storing operation

3) Order picking: This is the collection of activities to pick items ordered. Major decision problems in this operation include order batching, order picking, and routing. The order-batching problem is to decide how many and which orders should be picked in a batch (in other words, grouping of customer orders into pick lists). The order picking method in which a batch is comprised of a single order is called discrete order picking. In the meantime, the routing (or sequencing) problem in order picking operation determines the best pick sequence and the route of locations for the retrieval orders in a pick list.

4) Shipping: This operation assigns the product ordered to a shipping dock and schedules shipping trucks.

From an economical point of view, the order picking operation is most important because it constitutes about 55% of the total operating costs for a typical warehouse (Tompkins et al., 2003). But the efficiency of the order picking operation is closely related with operating policies on storing as well as order picking. One of the main issues in the order picking operation is the order batching problem. Order batching is to group line-items in several orders together in a single picking tour. Batching can be expected to reduce the average travel time per order by sharing a pick tour with orders. In the zone-based carton picking systems, we have to group and assign line-items within an order to cartons with limited capacity. We refer to this as *cartonization*. The cartonization essentially has the same characterization with order batching for grouping line-items into a carton to reduce the order picking cost by sharing a picking tour with line-items that are located in near slots. However the cartonization is different from order batching in that it groups line-items from the same order into cartons.

In traditional warehouse, there is a dependency between the slotting and order picking operations. Slotting operation has been performed efficiently using long-term demand so that the warehouse is not frequently replenished (i.e., yearly). Order picking operation (i.e., order batching, cartonization, and routing) is frequently performed based on the slot assignment of the SKUs by the slotting operation using the long-term demand. Due to the popularity of internet ordering systems and intelligent logistic and supply chain management systems, customers tend to order more frequently, in smaller quantities, and they require customized service. Companies tend to accept late orders while still needing to provide rapid and timely delivery within tight time windows (thus the time available for order picking is shorter). Turn-over rate of SKUs in the warehouse become short and diverse. Therefore, the determining of timing of the replenishment of distribution center and slotting of SKUs are not long-term decision and the warehouse operations become more complex and important to meet the dynamic demand trend. The distribution center in this dissertation is dynamically replenished specific SKUs for next pick wave, after pickers complete to retrieve all the SKUs for current pick wave. In this warehouse environment, the long-term demand is of limited value for the specific pick wave and the slotting operation and order picking operation have to determine simultaneously at the beginning of the pick wave. In the zone-based carton picking system in this dissertation, we face two primary assignment problems: assignment of SKUs to slots in the picking area (slotting); and assignment of line-items to cartons within an order (cartonization). The two primary assignment problems are interrelated with each other. In order to assign SKUs into slots efficiently, it is necessary to know that which line-items in an order are grouped together into the same carton. On the other hand, in order to assigning line-items into cartons efficiently, it is necessary to decide where SKUs are slotted and which SKUs are closely slotted together. This dissertation therefore deals mainly with two interrelating problems to reduce the order picking cost in a distribution warehouse as a part of order picking problem.

1.1 Problem statement

As stated earlier, this dissertation focuses on the order picking cost in a distribution warehouse. A typical distribution warehouse consists of two distinctive areas; forward picking area and reserve storage area. In the forward picking area, the items are stored and picked in SKUs (stock keeping units). Figure 1.2 shows the configuration of the forward picking area in a target distribution warehouse. An individual SKU is stored in a slot of the storage rack. The SKUs are replenished on a daily basis from the reserve storage area, which stores items in lots. This is an example of the warehouse under consideration in this study. This warehouse adopts the so-called zone-based picking system, which means the picking area is divided into several zones and an order picker is dedicated to each zone. In this warehouse, the turnover rates of items stored are so high that a picker is needed to serve a rack-face, in other words, a zone means a rack-face in this study. Since an order picker works for only one rack face, the routing problem, which is one of the main problems in order picking operation, is of little significance in this situation. There are two main decision variables to determine the order picking cost in the zone-based carton picking systems. The zone-based carton picking systems use cartons containing line-items to construct a single picking tour. Since the carton is directly shipped to a customer, the carton must contain line-items within a single order. Thus, the *cartonization* is one of decision variables to determine the order picking cost because it defines the assignment of line-items traveling a same picking tour and can construct a travel distance for pickings within a zone by referring given SKUs locations with the corresponding line-items. The cartonization becomes critical if the distribution warehouse has to ship large size orders being over-carton-capacity to the retail stores. In the zone-based carton picking system, assigning SKUs into slots in the racks within the zones in order picking area is called *slotting*.

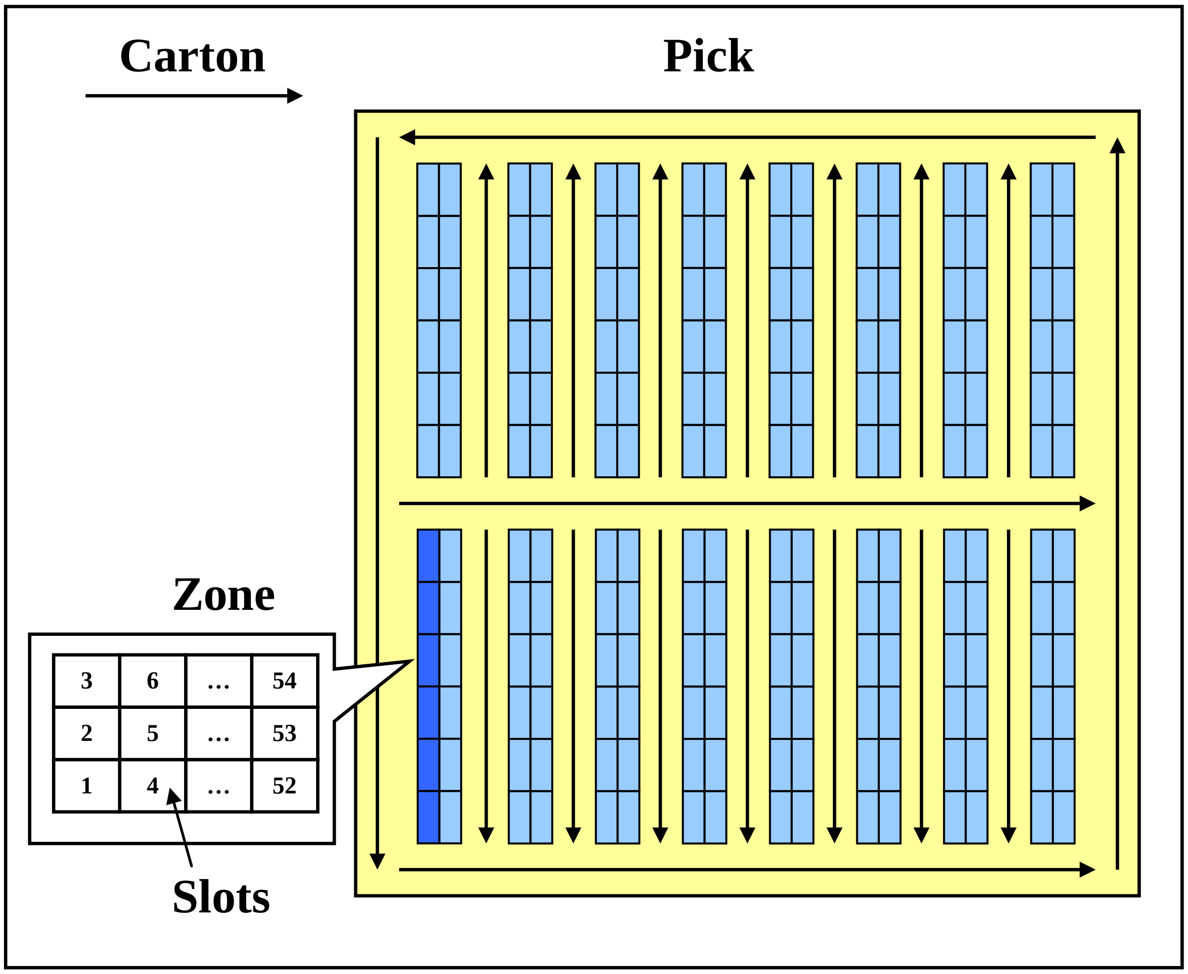


Figure 1.2 Picking area in a zone-based carton picking system

The slotting is the other decision variable to determine the order picking cost because it defines the assignment of slot locations of SKUs and it can construct a travel distance for pickings within a zone by referring given line-items in a carton with the corresponding SKUs.

In this warehouse, different sets of SKUs are picked on different days of the week and the picking area is re-slotted on a daily basis specifically for each pick wave. We called the warehouse environment as *dynamic whole warehouse replenishment environment*. In this warehouse environment, the long term SKU demand correlations are of limited use and the specific correlations in a given pick wave can be exploited to identify good slotting for the *specific pick wave*. In the dynamic whole warehouse environment, the order picking time is not able to construct in the zone-based picking system without both decisions for slotting and cartonization. The problems studied in this dissertation, therefore, are related with the slotting and cartonization operations affecting the order picking cost in the zone-based carton picking system. To clarify the configuration of the order picking system in this dissertation, we state several physical descriptions and operational descriptions that are valid for all the models to be proposed in the following chapters.

1) An order is comprised of a number of line-items. Each line-item in an order has a quantity (>= 1). Each line-item matches exactly one SKU in the picking area.
2) The slotting facility in the forward storage area is a set of equal-sized and double-sided racks. In each slot of the racks, a unique SKU is assigned. Each

SKU has an unique unit-volume respectively. We assume that each slot of the racks can contain total ordered quantity of the SKU in the pick wave. In other words, multiple slots cannot be assigned for a single SKU.

3) All cartons have the same fixed capacity. Since a carton must directly ship to a customer, cartons can contain only items for a single order

4) The SKUs in the order picking area are entirely replenished on a short-term periodic basis (i.e., daily basis) from the reserve storage area, which stores items in lots.

5) The cartons are transported between aisles and also through an aisle via an automated conveyor system (called pick-and-belt system). If the picker is working when an empty carton arrives at a zone, the carton waits at the zone initiation point until the picker completes the current job and returns to zone initiation point. To start picking process for a carton in a zone, the picker scans bar code on the carton so that the WMS (warehouse management system) can identify the carton. The time required to set up a zone at the beginning of a pick wave is called the *zone setup time*. The WMS uses a pick-to-light system and the slots corresponding to the line-items within the zone assigned to that carton are identified by a small light in front of the slot. The operator then walks down the aisle picking the specific quantity of each SKU with the "light on". The time required for identifying the slot locations of the line-items of the specific carton visiting the zone, which are assigned in the zone is called the

carton setup time.

6) Once an order picker completes the picking process for a carton, the carton is conveyed to the end of the aisle and is transferred to the next zone using a fast moving a conveyor system circulating zone-to-zone. The order picker returns to the zone initiation point to pick next carton waiting in the zone initiation point. Once a carton is completed in picking process visiting zones, it is loaded for direct shipment to the customer.

7) The picker's service time for a carton consists of carton setup time, the walking time, and the picking time. The first one is time to scan bar code on the carton and identify it. Since the picker always returns to the zone initiation point to pick the next carton, the *walking time* for a carton is double the walk-time from the zone initiation point to the farthest slot storing a SKU to be picked for the carton. The *picking time* depends not only on the number of slots to be visited, but also on the rack-levels (in other words, heights) of the slots.

8) The completion time of a picker in a pick wave is the sum of the zone setup time and the service times for all cartons assigned to the picker in the pick wave. Note that if we assume that there is a sufficient queue of cartons waiting at each zone initiation point so that the starved time is negligible, the pick wave makespan can be computed given these assignments. Since the target order picking system deals with a high quantity of cartons for a pick wave, this assumption appears reasonable.

In the zone-based picking systems, both balancing the zone-to-zone workload and improving the utilization of pickers among zones are important. To balance and improve the utilization of pickers in zone-based carton picking systems, we adopt that the main goal of this study is to minimize the pick wave makespan, which is defined as the maximum completion time over all the pickers in a pick wave. To minimize pick wave makespan in our warehousing system, both the slotting problem and the cartonization problem are important. This study proposes three optimization models to improve those two problems. First of all, under the assumption that the line-items in an order assigned to cartons with a limited capacity are known, a procedure that assigns the SKUs in the orders to slots is developed. Second model is to cluster line-items in an order into cartons with a limited capacity to reduce the pick wave makespan of pickers, when the slotting schedule (SKU-slot assignment) is given. Since one of the two variables is fixed in these two models, the results are sub-optimal. Therefore, we finally propose a model to solve the two problems simultaneously. To summarize, while the five sequential problems in Figure 1.3 can be included in the warehouse operation problems for the above-mentioned distribution warehouse system, this dissertation narrows down the interested research areas to slotting and cartonization problems.

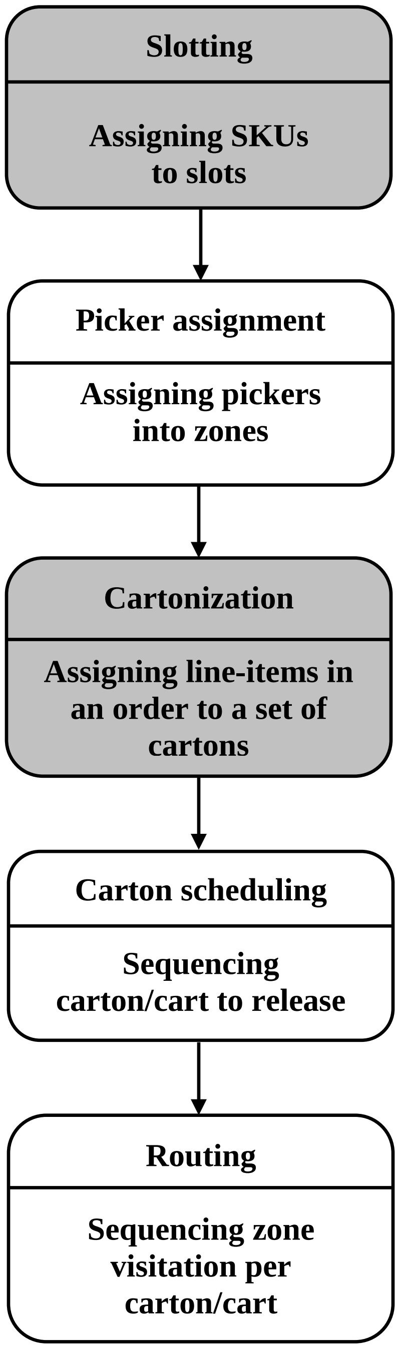

Slotting
Assigning SKUs to slots
Picker assignment
Assigning pickers into zones
Cartonization
Assigning line-items in an order to a set of cartons
Carton scheduling
Sequencing carton/cart to release
Routing
Sequencing zone visitation per carton/cart

Figure1.3 Warehouse operation problems in zone-based carton picking system

Although the first two approaches, in which the slotting problem (or the cartonization problem) is solved when the slotting (or cartonization) is given, have been extensively studied, we provide new approaches that are adapted to the situation in this study. For the third model that tries to solve the slotting and the cartonization problems simultaneously, there is no earlier study, and so we provide a new problem and its solution approach. From the studies, we can expect several contributions as follows:

Contribution 1: For slotting problem in the zone-based carton picking system, both SKUs individual popularity and the correlation between SKUs are important. In this dissertation, we develop a meta-heuristic using the simulated annealing method (SA-C). It improves the solution quickly based on a COI initial slotting solution. The proposed SA-C heuristic is relatively simple and provides a good solution of SKUs to specific slot locations using specific pick wave information in the zone-based carton picking system in the limited planning time.

Contribution 2: For the cartonization problem, we develop a cartonization heuristic algorithm. The cartonization essentially has the same characterization with order batching for grouping line-items into a carton to reduce the order picking cost by sharing a picking tour with line-items that are located in near slots. However, the cartonization is different from order-batching in that it groups line-items from the same order into cartons. There have been no relating studies on cartonization problem. The proposed heuristic algorithm has a relatively simple procedure using a classical bin-packing problem and slotting information of SKUs. Based on the cartoniztation heuristic, we can assign specific line-items in an order into cartons with a limited capacity to minimize pick wave makespan.

The performance of the proposed heuristic improves, as the number of line-items increases and the ratio of the mean order-volume to the carton capacity increases. The heuristics in this study shows a good performance consuming the reasonable number of cartons compared to the number of cartons using classical bin-packing problem.

Contribution 3: Under dynamic whole warehouse replenishment environment, independent solutions of the previous two problems (i.e., slotting and cartonization) result in a sub-optimal solution. Thus, we must deal with solving the both problems simultaneously to avoid the local optimum solutions in two sub-problems. To solve the slotting and cartonization problems simultaneously, this study proposed iterative heuristic solution approach. Using the previous heuristics in Chapter 3 and 4, this heuristic iteratively solved the slotting/cartonization heuristic in current stage based on the previous solution of cartonization/slotting in the previous stage. The method we developed for generating artificial correlated data is a contribution.

Contribution 4: We developed the first random pick wave generating method reflecting the correlation between SKUs. In multiple picking, the slotting method is highly dependent upon the correlation between SKUs in a pick wave. This method provides the effect of the correlation between SKUs on the performance of the slotting methods by controlling the number of correlated SKUs per each specific SKU and the strength of the correlation.

1.2 Scope of the study

The rest of this dissertation is organized as follows. Chapter 2 discusses a comprehensive literature on the warehouse operation problems. In Chapter 3, a mixed integer programming (MIP) model and several heuristic algorithms for slotting problem are provided, while the cartonization problem for given slotting schedules are studied in Chapter 4. Chapter 5 proposes an iterative heuristic approach for the combined problem of both the slotting problem and the cartonization problem. The approach is based on the methodologies proposed in Chapter 3 and 4. The structure of the problems in this study is depicted in Figure 1.4. Chapter 6 ends this dissertation with some concluding remarks and future research directions.

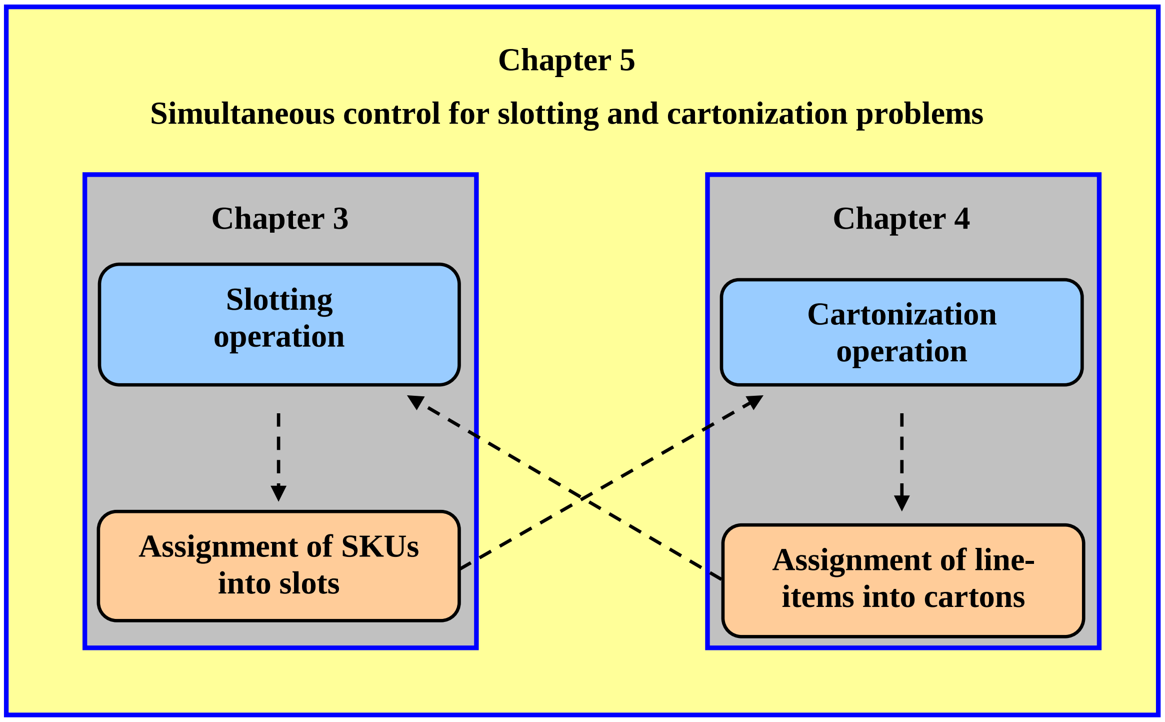

Chapter 5
Simultaneous control for slotting and cartonization problems
Chapter 3
Slotting operation
Assignment of SKUs into slots
Chapter 4
Cartonization operation
Assignment of line-items into cartons

Figure 1.4 Scope of the study

Chapter 2
Comprehensive Literature Review on Warehouse Operations

2.1 Introduction

Warehouse management problems are classified into three categories: warehouse location, warehouse design, and warehouse operation. The warehouse operation problems are the major focus of the dissertation. In this chapter, therefore, we present a literature review on the warehouse operation problems. As stated earlier, the warehouse operation problem can be classified into four areas according to the main operations of a warehouse: receiving, storing, order picking, and shipping. Figure 2.1 describes the typical functional areas and material flows within warehouse. The receiving operation includes the unloading of products from the transport carrier, updating the inventory records, inspecting to find if there is any quantity or quality inconsistency. Then it is transferred to the reserved area for pallet picking or the forward area for case picking and to the broken case picking or to the directly cross-dock area in shipping area. The storing operation includes indentifying an appropriate location in the slotting area and storing items for future picking. It can be included as a full pallet into reserved area or as a case into case picking area and individual small items into broken case picking area. The main issue of the slotting function is to find the method of slotting to effectively support future retrieval. The order picking operation is labor intensive and expensive and is the primary component of warehouse operations.

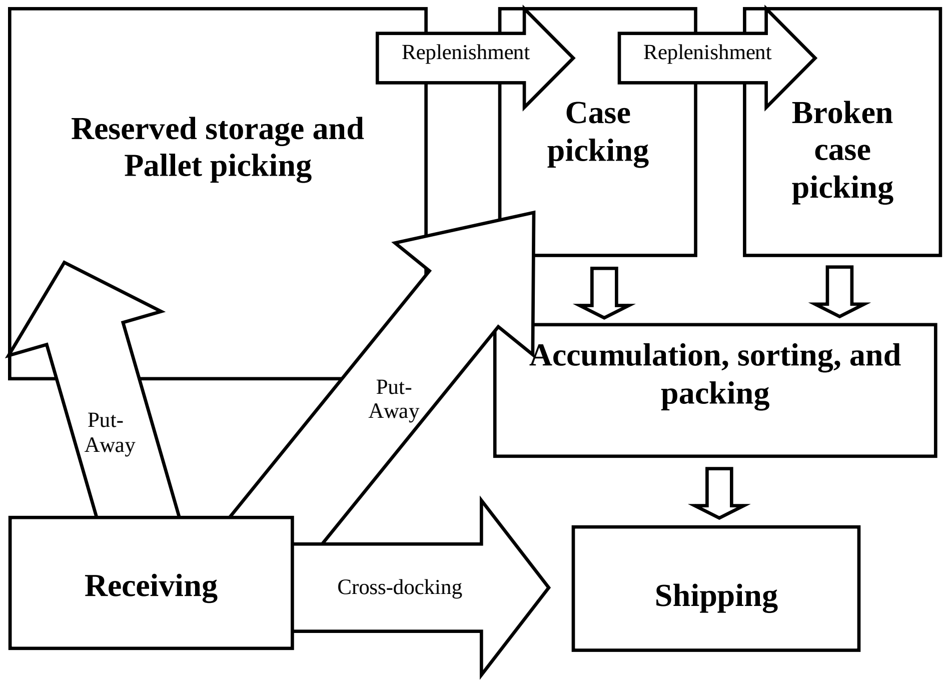


Figure 2.1 Typical warehouse operations and material flow (Tompkins et al., 2003)

Figure 2.2 shows the order picking cost is estimated to be as much as 55% of warehouse operating costs (Tomkins et al., 2003) and Drury (1988) and Coyle et al. (1996) also reported that it is estimated about 65% of the total operating costs for a typical warehouse. Order picking involves the process of picking products from slotting area to reflect a set of customer orders. It also includes order batching, assigning pickers into zones, the routing of pick-device or pickers. The shipping operation is the last operation in the warehouse. It determines shipping dock for arriving items from order picking area and controls cross-docking operation when the received products are transferred directly to the shipping docks. In this chapter, we review earlier researches classified by the four main warehouse operations. Some other review papers dealing with the warehouse operation problems can be referred (Wascher, 2004; Gu et al., 2007; De Koster et al., 2007).

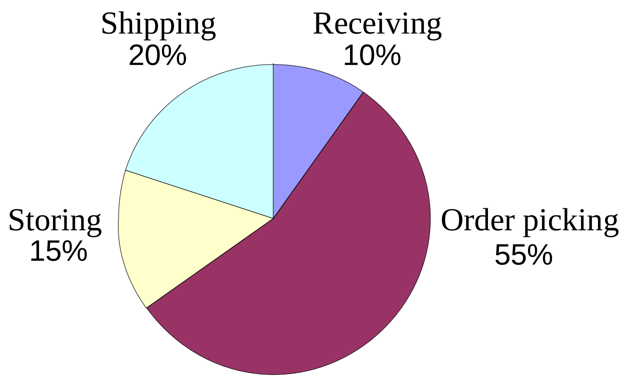


Figure 2.2 Typical warehouse operations cost (Tompkins et al. 2003)

2.2 Receiving and shipping operation

The receiving operation is a set-up operation for all other warehouse operations. It includes unloading products from the transport carrier, updating the inventory records, inspecting the inventory to find if there is any quantity or quality inconsistency. Then it is transferred into traditional put-away areas or cross-docking area. The traditional put-away areas indicate the reserved storage area for the pallet-picking or the forward-picking area for the case-picking and the broken case picking store. For cross-docking areas, received products are sent directly from the receiving docks to the shipping dock. The cross-docking area requires the ability to schedule inbound loads to match outbound requirement on a daily or even hourly basis. In addition to the balancing of personnel, docking doors, and staging space are also necessary for efficient shipping.

The shipping operation is the last operation of warehouse (chronologically). Shipping operation should be performed within a limited shipping staging area. Shipping dock management is important for steady outbound load shipping control. Outbound truck shipping dock-loading scheduling should be done before picking items into shipping area. The research on shipping has been focused on the truck-to-dock assignment problem. In general warehouse, the number of receiving-docks and shipping-docks is not fixed, because it can be dynamically controlled by receiving and shipping-waves arriving into warehouse during a day.

2.3 Storing operation

In general, three fundamental decisions are introduced for the storing operation (i.e., how much inventory should be kept for a SKU (stock keeping unit) in the warehouse, how frequently or when should the inventory for a SKU be replenished, and where should the SKU be stored in the warehouse). The first two decisions belong to the traditional inventory control area. In this section, we only focus on third decision. We called this slotting. The slotting method is the rule based on which SKUs are assigned into slots to optimize the warehouse objectives. The objectives of slotting operations usually involve either maximizing resource utilization while satisfying customer requirements or minimizing material handling cost subject to resource constraints. The basic decisions, propositions, and constraints in slotting operations can be described in Table 2.1.

Table 2.1 Basic decisions, given information, and constraints in slotting operation

Decisions:	Given information:	Constraints:
Assigning SKUs into the storage location (slotting)	Physical configuration and storage layout Storage locations with dimension and sizing The set of SKUs to be stored Demands and order quantity, arrival and departure time of orders	Storage area capacity The utility of pickers based on the picking ability of pickers

2.3.1 Dedicated slotting policy

In dedicated slotting, each SKU is permanently assigned a dedicated slot (or set of slots). A major disadvantage of the dedicated slotting method is that space utilization can be quite low in dedicated storage environments as space must be allocated for the maximum inventory level of all SKUs regardless of their actual inventory levels. An advantage of this slotting method is that human order pickers become familiar with SKU locations and this familiarity can save both slotting and picking time in the warehouse. This slotting policy can save work because the items can be logically grouped and assign the slotting area. If there are special products (i.e., heavy, fragile, or risky products), the dedicated slotting is often appropriate considering product characterization.

2.3.2 Random slotting policy

In a random slotting policy, slots for incoming SKUs are assigned in a completely random manner. That is, an incoming SKU will be assigned any available slot with equal probability. High space utilization and ease of slot selection are the primary advantages of the random slotting method. In randomized slotting, however, it can be hard to find the locations of retrieval SKUs during the picking process (Choe and Sharp, 1991), and the use of a computer-controlled warehouse management system (WMS) is generally required. If product storing employees choose the slot for storage of SKUs, then they will generally choose the closest empty slot. The slotting method is that the first empty slot encountered by an employee is chosen as the slot for a storing SKU. This slotting decreases travel-distance, however, it is concentrated to slots fully around the depot and gradually more empty towards the back if there is excessive warehouse containing capacity. This can lead to blocking and congestion during picking. Hausman et al. (1976) argued that the closest open location slotting and random slotting have a similar performance if products are moved by full pallets only.

2.3.3 Full-turnover based slotting policy

This policy distributes items over the storage area according to their turnover. In the full-turnover based policy, the items with the highest demand are assigned to the easiest accessible slot locations and the items with low demand are assigned to somewhere towards the back of warehouse. One of the most popular types in the dedicated slotting policy is Cube-per-order index (COI) storage assignment, where the COI of an item is defined as the ratio of the required storage space to the order frequency of the item (Heskett, 1963, 1964, Kallina and Lynn, 1976, Malmborg and Bhaskaran, 1987, 1989, 1990, and Malmborg, 1995, 1996). The COI-based slotting method sorts items by increasing COI ratio and sorts locations on increasing distance from the I/O point. Next, items are assigned one by one to locations in this sequence (items with the next lowest COI ratio to next quickest-to-access locations). The first reported COI-based storage assignment is given in Heskett (1963, 1964). Then, many authors have emphasized on his work under different picker travel operation policies (Caron et al, 1998, Petersen and Schmenner, 1999, Hwang et al. 2004). Harmatuck (1976) and Kallina and Lynn (1976) proved the optimality of COI for single command traveling operation. Malmborg and Bhaskaran (1987, 1990) proved the optimality of COI for dual command traveling operation in unique and non-unique layout. Malmborg and Bhaskaran (1989) demonstrated the order picking cost optimality of the COI if vehicles are routed to execute multiple commands in single-aisle traveling operation. The main disadvantage stems from the dynamic change of demand rates and SKUs in the warehouse. In COI slotting policies, re-slotting is periodically required due to changes in the SKU order frequencies. If the SKUs assortment changes too fast to build the slotting of SKUs, reliable demand statistics may not be expected. In this case, the COI-based slotting is not effective. (De Koster et al., 1999).

Volume-based (frequency-based or turn-over based) storage assignment is the other type of the dedicated storage assignment method. It is studied by, for example, Petersen (1997, 1999, 2000), Petersen and Schmenner (1999), Petersen et al. (2004), and Petersen and Aase (2003). This method assigns items to storage locations according to their (expected) pick volume and it usually locate the item with high pick volume closest to the I/O point. The pick volume of an item can be expressed in the number of units or pick lines during a certain time horizon. The difference between this method and COI-based storage is that the volume-based assignment only considers the popularity of items without considering their space requirements for individual items.

2.3.4 Class-based slotting policy

Class-based slotting is adopted from the idea of Pareto's method in inventory control. The basic idea of the Pareto's method groups items into several classes and the grouped inventories are controlled differently. In class-based slotting, the fast moving class contains only about 15% of the items stored but contributes to about 85% the turnover. This method assigns items to storage locations based on item class. It divides both items and storage locations into an identical number of classes. Item classes are based on turnover rate. The item classes are sorted on decreasing turnover rate and the storage location classes on increasing travel distance from the I/O point. Next, the item classes are sequentially assigned to the storage location classes (which should be large enough to contain the SKUs) in this sequence. Within a storage class, items are randomly stored. The major difference between this method and the volume-based assignment method is that this method assigns items to storage locations based on a class basis, while the volume-based method uses an individual basis. In general, the number of classes is restricted by three.

Most of the research on the class-based storage has been performed for AS/RS systems. Firstly, Hausman et al. (1976) considered the problem of finding class regions for an AS/RS using the class-based storage assignment method with single traveling operation. They proved that L-shaped class regions where the boundaries of zones accommodating the corresponding classes are square-in-time and are optimal, minimizing the mean single-command travel time. Starting from this study, a number of papers on class-based storage are studied in AS/RS (Graves et al., 1977 and Rosenblatt and Eynan, 1989, etc.). In low-level aisle of picker-to-part systems, there are various possibilities for positioning the class A, B, and C. Jarvis and McDowell (1991) suggested that each aisle should contain only one class, resulting in within aisle storage. They compared random slotting and several COI-based class-based slotting policies based on different ABC inventory curves in a rack based slotting area. The results showed that the class-based slotting decreases more travel time than a random slotting. But their research is limited in that it assumes that the aisles only allow one-way travel and are limited to traversal routing. Petersen (1999, 2002) and Petersen et al. (2004) compared multiple configurations of pick-and-walk order picking systems with across aisles. Roodbergen (2004) compared various slotting methods for warehouse layouts with multiple cross aisles. Le-Duc and De Koster (2005) optimized the storage-class positioning. They claimed that the slotting with across aisles is close to optimal. De Koster et al. (2007) from the literature review paper concluded that there is no firm rule to define a class partition in lower-level picker-to-part order picking systems.

2.3.5 Correlated slotting policy

None of the previous slotting assignment policies mentioned above consider the relation between items. Sometimes in practice, the correlation between items is important to assign SKUs into slot area to pick efficiently for customer orders. For example, customers may tend to order an item with other related items. In this case, the correlated items should be assigned to closer slots to reduce travel time. The main issue of correlated slotting policy is to locate similar items in the same region of the storage area. To do this, the statistical correlation between items should be known and predictable. Frazelle and Sharp (1989) and Frazelle (1990) developed a procedure to assign items to locations based on the correlation between items. This approach recognizes that items that are likely to appear in the same order should be stored in nearby locations. Brynzer and Johansson (1996) developed a heuristic for slotting problem emanating from the product structure. Manzini (2006) developed three order clustering heuristic rules based on a strategy of correlation between SKUs in picker-to-part order picking systems using the correlation index from Frazelle and Sharp (1989).

In complementary-based correlated slotting method, two major phases are performed. In the first phase, it clusters the items into groups based on a measure of strength of joint order such as the correlation between items. In the second phase, it assigns items within the cluster and the next cluster assigned close to the previous cluster. Rosenwein (1994) showed that the clustering problem can be formulated as a *p*-median problem. For finding the position of clusters, Liu (1999) suggests that the item type with the largest demand should be assigned to the location closest to the depot (volume-based strategy), while Lee

(1992) proposed a new heuristic of slotting problem in a man on board AS/RS with multi-address picking. In this heuristic, he considers both order frequency and order structure. He clustered the items and assigned the cluster by COI-based slotting method using the space requirement as an initial slotting assignment of SKUs and then perform improving search by using the pairwise interchange of SKUs. The second type of correlated slotting is called the contact-based method. This method is similar to the complementary method, except it uses contact frequencies to cluster items. The contact-based method is considered, for example, in Van Oudheusden et al. (1988) and Van Oudheusden and Zhu (1992).

In zone-based batch picking systems, Jane and Laih (2005) proposed MIP model for assignment problem of items to zones and developed on items-to-zone assignment heuristic to balance the workload among all pickers using the correlation list in a synchronized zone order picking. Peters and Smith (2001) and Smith and Kim (2008) proposed the assignment for specific SKUs to slots in the zone. Peters and Smith (2001) paper served as the initial inspiration for this dissertation. They proposed the COI-based initial slotting and then improved the initial solution using the correlated slotting (CS) improvement search method. Smith and Kim (2008) compared the performance of correlated improvement with COI slotting using artificially generated correlated carton list.

2.3.6 Dynamic slotting policy

Most of the literature related in slotting and order picking assume that the slotting location of SKUs is dedicated or random in static warehouse environment. More intelligently the warehouse uses a long-term demand historical data for slotting. The storage location assignment problem (SLAP) problem in the literature has mostly used a static demand (i.e., it assumes that the incoming and outgoing SKUs flow patterns are stationary over the planning horizon). In some cases in reality, the patterns of SKUs changes dynamically due to factors such as seasonality and the life-cycle or turnover rate of the SKUs. Therefore the slotting location of SKUs should be controlled to reflect changing products flows. We call this as *dynamic slotting*.

There are two types of the dynamic slotting. The first type of the dynamic slotting is the dynamic partial slotting of warehouse. In the dynamic partial slotting, each SKU in the warehouse has different turnover rate. Therefore, only some SKUs which have out of stock for next pick-wave should be replenished at the end of the current pick-wave. The SKUs that have inventories in the slot should be relocated to other SKUs for an efficient slotting of the next pick-wave. Therefore, two movements of SKUs are potentially required in the dynamic partial slotting (i.e. the replenishment movement of SKUs from the reserved area to the forward area and the relocation movement of slot from the rest SKUs after picking the current pick-wave). The relocations of SKUs within forward picking area are only beneficial when the expected savings in order picking outweighs the corresponding relocation cost. Therefore, decisions must be made concerning which a set of items to be relocated, where those to be relocated, and how to schedule the relocations. In the partial slotting, the decisions for relocation must be carefully executed concerning which set of SKUs to be relocated, where to relocate them, and how to schedule the relocations. The replenishment planning problem from the forward to the reserved area has been studied in Hackman and Plazman (1990), Frazelle (1994), Van den berg et al. (1998), and Bartholdi and Hackman (2008). In these studies, the main objectives are to decide how much of each SKU is placed in the forward picking area and where areas in which a single SKU can be stored and picked, depending on the storage and pick quantity under the restricted small forward picking area. In relocation of SKUs within the forward picking area, Christofides and Colloff (1972) studied finding the optimal ways of rearranging items in a warehouse from their initial positions to their desired final locations. The authors proposed a two-stage algorithm that produces the sequence of item movements necessary to achieve the desired rearrangement and incur the minimum cost spent in the rearranging process. Roll and Rosenblatt (1987) described the situation when the storage area is divided into separate zones and any incoming shipment must be stored within a single zone. It might happen that none of the zones has sufficient space to accommodate an incoming shipment. In this case, it is advisable to free some space in a certain zone to accommodate the incoming shipment by shifting some stored products in that zone to other zones. Muralidharan et al. (1995) proposed the shuffling algorithms that the set of high-demand items are relocated to the near I/O point to minimize the total relocation cost, when the stacker crane is idle in AS/RS. Two shuffling algorithms are proposed named shuffling with nearest neighbor heuristic (SNN) and shuffling with insertion (SI). Both algorithms first define relocation arcs. Then the relocation route of the stacker crane is determined. In the SNN heuristic, an arc i with minimum distance, in terms of travel time, between the I/O and the beginning node of arc i is chosen as the first arc to travel. Then, another arc j with the minimum distance between its beginning node and the ending node of arc i is selected as next arc to travel. In the SI heuristic, an arc i from the unsequenced arc set is chosen that is closest to the I/O point first. Then the heuristic chooses another arc j from the unsequenced arc set that is nearest to the head of previously chosen arc i and arc j is inserted before it. The time to cover the arc sequence (from arc j to arc i), and its reversed sequence (from arc i to arc j) is calculated. The arc sequence (from arc j to arc i) is included in the tour route if the time to cover this arc sequence is less than the time to cover its reversed sequence. Otherwise, the reversed sequence (from arc i to arc j) is included in the tour. The heuristic is repeated until all the arcs in the unsequenced set are exhausted or the time to travel these arcs becomes greater than the idle time. Jaikumar and Solomon (1990) determined the products to be relocated and their destinations with the objective to find the minimum number of relocations that result in a throughput satisfying the throughput requirement in the following busy periods.

The second type of dynamic slotting is the dynamic whole slotting of warehouse. In this environment, the number of SKUs and their quantities in current pick-wave should be determined. After order picking process for the current pick-wave, the forward picking area is emptied. The warehouse should then replenish the SKUs into the whole forward picking area for the next pick-wave. In this case, the reslotting procedure is performed at the end of the turnover. The main decision of the dynamic whole slotting is to select SKUs into the forward pick area from the reserved area and where the selected SKUs are slotted. Goetschalckx and Ratliff (1990) studied a shared slotting policy for a unit load warehouse where over time different SKUs are stored in the same storage slot. Their work was focused on the fact that individual unit loads of the same SKU will stay in the storage area for different amounts of time. Thus, the shared storage policy tries to exploit the difference between products in terms of inventory profiles and usage patterns. Landers et al. (1994) and Sadiq et al. (1996) also investigated the problem of reslotting SKUs over time. Under less than unit load picking, they consider dynamic environments where the products evolve through a life cycle and thus the product mix varies over time, which creates a need to resize SKU slots and reassign the SKU locations. Their procedure addresses a wide range of issues related to this reassignment problem. Part of their procedure includes a clustering algorithm that attempts to determine which SKUs should be stored together based on their long-run average correlation. The paper tested the performance of these procedures but doesn't provide details of the clustering algorithms used. Kim and Smith (2008) proposed an efficient slotting mythology under dynamic whole warehouse replenishment environment. Using the correlation among SKUs per pick-wave in zone-based order picking systems, they proposed the correlated slotting improvement heuristic, in which it assigns the correlated SKUs to the near to each other based on a COI initial slotting. It shows almost 20% improvements under high correlated orders than the COI based slotting policy.

2.4 Order picking operation

The order picking operation is the most labor intensive operation in the warehouse. The primary goal in the order picking systems is to pick orders accurately and efficiently before they are sorted/packed and shipped for delivery to the customer using minimum number of labors or cost. To resolve the goal of order picking systems, a variety of literatures are focused on the problem. In this section, we classified the order picking problem into three picking types: single order picking, batch order picking, and zone order picking and we reviewed comparative studies for factors affecting the performance of order picking operation. At the end of this section, we reviewed on packing algorithms. The packing operation is usually performed after order picking operation. However, the packing process should be performed during picking in the target carton picking systems, because the cartons after order picking process ships directly to customer. The planning of items packed together must be finished at the beginning of the order picking operation. Therefore, we assign the packing operation into one of sub-operations of the order picking operation and one of key factors to determine the efficient order picking cost in this dissertation. While there are many good studies of the routing operation (Ratliff and Rosenthal, 1983, Hall, 1993, Peterson, 1997, Roodbergen, 2001, Roodbergen and De Koster, 2001, De Koster et al., 2007) we do not study this work because the routing decision is not on issue in the target environment.

2.4.1 Single order picking

Single order picking in industry is popular picking method which comprises of single or double-deep pallet racks. In single order picking, each order picker completes one order at a time. If SKUs are palletized and unitized, this warehouse is called *aisle-based unit-load warehouse*. The major advantage of single order picking is that picking is simple and order integrity is never jeopardized. The major disadvantage is that the order picker is likely to travel over large portion of the warehouse to pick a single order.

There are several reasons for few literatures found, even if the single order picking in aisle based warehouse system is popular in practice. First, it is easy to control the picking process once a storage assignment is given. Second, it is a special case of batch picking if each picking tour has only one pick. Most of the papers in single order picking with single command and dual command are focused on an analytical expected travel time model for a given warehouse design (Francis, 1967, Bassan, 1980, Larson et al. 1997, Pohl et al, 2009b). Recently, Gue (2006), Gue and Meller (2008), and Pohl et al (2009a) studied a unit-load warehouse picking system with non-horizontal and vertically aligned aisles.

2.4.2 Order batching

A second order picking policy for order picking is batch picking. When orders are small, there is a potential benefit for reducing travel times by picking a set of orders in single picking tour. Thus an order picker picks a number of orders (a batch of orders) during his picking tour. The major advantage of the batch picking is reduction in travel time per item. The disadvantages of the batch picking are the time required to consolidate the items into customer orders and the potential for picking errors. Orders are consolidated in two different ways. First, the order picker uses separate containers to sort line-items of different order during picking tour (sort-while-pick). Second, the line-items and quantities of different orders are picked together and the orders are sorted after picking (pick-and-sort). The general objective in order batching in aisle-based order picking systems is to minimize travel time to pick line-items in all orders. Gademann et al. (2001) considered the maximum batch travel-time among batches. Meanwhile, Gademanne et al. (2005) and Bozer and Kile (2008) considered their objective as minimizing total batch travel times. If a zone picking system is employed under batch picking, the picking time among zones should be balanced during pick-wave or specific time window to improve the overall productivity of zone-based picking systems (Jane and Laih, 2005, DeKoster and Yu, 2008, Kim and Smith, 2008). Several studies proposed MIP formulations in manual aisle based order picking systems. Hwang and Kim (2005) also measured the similarity between orders by three types of routing policies in low-level order picking systems with front and back cross-aisles and P/D point located in the most left-point in the front cross-aisle. Both studies developed clustering models using MIP programming to maximize the total association of batches. Bozer and Kile (2008) formulated MIP model minimizing sum of batch traveling distance in low-level order picking systems with front and back cross aisles and P/D point located in the center-point in the front cross-aisle. In synchronized zone-based batch picking systems, Parikh and Meller (2006) proposed MIP model maximize total number of items fulfilled. There is no literature to formulate MIP model on catonization problem minimizing pick-wave makespan of pickers in zone-based carton picking systems. The basic decisions, propositions, and constraints in the order batching problem can be described in table 2.2.

Table 2.2 Basic decisions, given information, and constraints in order batching operation

Decisions:	Given information:	Constraints:
Grouping orders for assignment to picking devices or picking resources	Warehouse configuration	Capacity of picking resources
	A set of orders to pick during a shift or a pick-wave	Picking shift time
		Order or pick-wave due-date
	Information of SKU-slot	Picking time balance of pickers
	Pick-wave schedule	

Choe and Sharp (1991) classified two criteria for order batching: the proximity of pick location and the time windows for picking. Proximity batching assigns each order to a batch based on proximity of its storage location to those of the order. The main issue in proximity batching algorithm is how to measure the proximity metric among orders, which implicitly assumes a pick sequencing rule to visit a set of locations. Wascher (2004) classified the proximity batching proposed by Choe and Sharp (1991) into three types of heuristic algorithms such as priority rule-based algorithm, seed algorithm, and savings algorithm. Table 2.3 presents a summary of the literature on various criteria of the order batching and their algorithms.

Table 2.3 Order batching criteria

Order batching criterion:	Algorithm:	Example
Proximity batching	Priority-rule algorithm	Gibson and Sharp (1992)
	Seed algorithm	Elsayed (1981) Elsayed and Stern (1984) Elsayed and Unal (1989) Gibson and Sharp (1992) Hwang and Lee (1988) Hwang et al (1988) Pan and Liu (1995) De Koster (1999)
	Time saving algorithm	Rosenwein (1996) Hwang and Lee (1988) Elsayed and Unal (1989) De Koster et al. (1999)
Time window batching	Tardiness or lead time	Comier (1987) Elsayed et al. (1993) Elsayed and Lee (1996) Won and Olafsson (2005)

In priority rule-based algorithms, an initial priority is assigned to each customer order. Then, in the order given by the priorities, the customer orders are assigned one by one to batches until the capacity constraint is violated. Several methods have been suggested for the priority rule-based algorithm. The most straightforward method is the first-come-first-serve (FCFS) rule. Gibson and Sharp (1992) suggested two-dimensional and four-dimensional space-filling curve and mapped the coordinates of the locations of the items of a customer order into a value on the unit circle. Bin-packing methods are the other class of the priority rule-based algorithm. Next-fit (NF) batches are completed with in the sequence given by the priorities. When the addition of another customer order is performed, a new batch is started if the batch violated the capacity constraint. In first-fit (FF) method, batches are numbered in the sequence in which they are started. Then the current customer order is assigned to a batch with the smallest number into which it fits. Best-fit (BF) method grouped batches into which a customer order would fit. Then it is assigned to the one where the batch leaves the smallest remaining capacity. In the second, seed algorithm methods generate batches sequentially (i.e., a new batch is not started before the current one has been closed). In order to construct a batch, an order is selected as the so called “seed” of the batch. Succeeding orders following the seed order are added to the batch until the capacity of the batch is exhausted. Elsayed (1981) and Elsayed and Stern (1984) have developed the seed algorithm and applied in AS/RS. In manual aisle based warehouse, De Koster (1999) systematically proposed several seed-selection rules (i.e., selection of a random order, an order with the largest number of positions, an order with the longest picking tour, and an order with the largest aisle length, etc). The seed-selection rule can be applied in two ways. Under single model, the originally selected customer order only serves as the seed for the present batch. Meanwhile, in cumulative mode, all customer orders already assigned in the current batch make up for the seed of the batch. The order-addition rule determines which an unassigned order should be the next one to be added to the current batch. In this rule, an order having a minimum proximity with the seed is selected into the current batch among unassigned orders. Usually, an order is selected whose “distance” to the seed of the current batch is minimized. The distance between an unassigned order and the seed can be defined in several ways such as the sum of the travel distances between every location of a seed item and the closest location of any item in the order, the sum of the travel distance between every location of an item in the order and the closest location of any item in the seed, the number of additional aisles which have to be visited if the order would be added to the seed, and the difference between the gravity centre of the seed and the gravity centre of the order, etc., (De Koster, 1999). As the last algorithm, Savings algorithms are based on the well-known Clarke-and-Wright (C&W) algorithm for the vehicle routing problem (Clarke and Wright, 1964).

In time window batching, Won and Olafsson (2005) used customer response time by jointly considering the batching and picking operations. Usually the time window may be fixed or variable. Tang and Chew (1997), Chew and Tang (1999) and Le-Duc and De Koster (2003, 2007) considered variable time window order batching (i.e. number of items per batch is fixed) with stochastic order arrivals for manual order picking. They model the problem as a batch service in queuing model. For each possible picking batch size, they first estimate the first and second moments of the service time. Then using the first and second moments, they can find the time in systems of a random order. Finally the optimal batch size is then determined. Simulation model was then compared with the analytical stochastic model. Comier (1987) proposed a heuristic for batching and sequencing orders to minimize the weighted sum of order picking time and tardiness in an AS/RS. Elsayed et al. (1993) and Elsayed and Lee (1996) considered the order batching problem in a man-aboard order picking system with minimizing the penalties and the tardiness of orders. They proposed a heuristic which first establishes batches and then determines the release times for the batches. The main issue of the seed algorithm and the combination rule in the savings algorithm and in the proximity batching is to how closeness metric is defined between orders for adding an order into batch. Gu et al. (2007) classified the order batching studies into a various closeness metrics. We also summarized the closeness metrics for batching and related literatures in Table 2.4.

Table 2.4 Closeness metrics for batching and related literatures

Closeness metrics:	Literatures (metrics used)
1. Number of common locations between two orders	Chrisman (1976,1977) (10)
2. Combined number of locations of two orders	Elsayed (1981) (1)
3. Sum of the distance between each location of one order and the closest location on the other order	Elsayed and Stern (1983) (1,2,3) Elsayed and Unal (1989) (6)
4. Difference of the order-theta values of two orders defined based on space-filling curves	Gibson and Sharp (1992) (3, 4) Hwang and Lee (1988) (8)
5. The number of additional aisles to travel when two orders are combined	Hwang et al. (1988) (9) Pan and Liu (1995) (1,3,4,6,8)
6. Savings in travel when two orders are combined	Rosenwein (1996) (5,7)
7. Center of gravity metric	De Koster (1999) (3,5,6,7)
8. Economic convex hull based metric	Gademann et al (2001, 2005) (10)
9. Common covered regions or areas	Chen and Wu (2005) (11)
10. Travel time	Hwang and Kim (2005) (12)
11. Association between orders	Bozer and Kile (2008) (10)
12. Routing or geographic region similarity	Ho et al (2006, 2008) (12)

Chisman (1975, 1977) presented two heuristics for the order batching problem by considering vehicle routing problem. Hwang and Kim (2005) measured the proximity of the similarity between orders to three routing policies. They include the similarities into p-median clustering integer programming formulation for order batching. They also suggest a heuristic clustering algorithm. The majority of literature has been focused on the objective of minimizing the total order picking time of batches. In practice, there might be other important criteria, for example, lead time and tardiness of shipping due-date. This criterion is called as time-window batching. In this batching method, the orders arriving in the same time interval or window are grouped as batch. Several studies are grouped into a set of orders and pick-devices by the order due date or by the penalty of violating the due-date. Chen and Wu (2005) measured the similarity between orders by taking into account the level of association between orders in order picking systems with front and back cross-aisles and P/D point located in the most left-point in the front cross-aisle. They develop a clustering model based on 0-1 integer programming to maximize the total association of batches. Hsu et al. (2005) developed genetic algorithm to solve batching problem.

2.4.3 Zone-based order picking

The previous two picking policies are defined that the order picker picks line-items in whole picking area. Zone-based order picking divides order picking area into zones. Each order picker is assigned to pick the part of order that is in his assigned zone. The zone-based order picking problem has received little attention despite its important impact on the performance of order picking systems. The basic decisions, propositions, and constraints in order batching problem are described in Table 2.5.

The major advantage of zone-based order picking is that travel congestion is reduced because each order picker is assigned to pick a part of the order. In addition, the order picker assigned to a small zone is familiar with item locations in the zone and picking time for a batch is reduced because line-item is separated by zones. The major disadvantage of zone-based order picking is that orders are split and must be consolidated again before shipping.

Table 2.5 Basic decisions, given information, and constraints in zone-based picking operation

Decisions:	Given information:	Constraints:
Assigning zone to pickers. Assigning zone to SKUs	Warehouse configuration SKUs information to be stored	Utility of pickers Slots size in a zone Balance of picking time of pickers

Two types of zone order picking systems can be used. The first zone-based picking system is pick-and-pass system. Using this system, one order picker starts on an order (or batch of orders) and, when he finishes his part of line-items of an order (or batch of orders), the carton containing the line-items and pick list passes over to the picker in the next zone. Once the carton containing an order (or batch of orders) visits all relevant zones where the line-items are included, it has finished picking. Carton pick-and-belt picking eliminates the consolidation procedure. In this picking procedure, a carton is assigned an order or a part of order and travels the zones in which SKUs in the order are slotted. After the carton finishes picking the SKUs, it is directly shipped to the customer. The second zone-based picking system is parallel (or synchronized) picking, where a number of order pickers located in their zone start picking operation of the same order. The partial orders are merged after picking.

De Koster (1994) developed an analytical model for a zone-based pick-to-belt order picking systems using a Jackson queuing network which allows rapid estimation of order throughput times and average work-in-process. He compared the analytical results with simulations. Recently, Yu and De Koster (2008) proposed an approximation model based on G/G/m queuing network modeling using Whitt's queuing network analyzer to analyze pick and pass order picking systems. The pick-and-pass system proposed is also decomposed into conveyor segments and pick stations like the study on De Koster (1994). Then the decomposed conveyor segments have a constant processing time, whereas the service times at a pick station depend upon the number of line-items in the order to be picked at the station. Based on the analytical model, Yu and De Koster (2009) studied the impact of order batching and zone size on the mean order throughput time. They found an optimal batch size is always exists and the batch size has large impact on mean order throughput time.

Petersen (2000) mentioned that the choice of a picking strategy can have a tremendous effect on the efficiency and the cost of a picking system in mail order companies. To this end, he evaluates five order picking strategies: discrete (or strict), batch, sequential (or pick and pass) zone, simultaneous zone (which he calls batch-zone), and simultaneous zone-wave using a simulation model. Based on the results, he concludes that simultaneous zone-wave picking and batch picking are superior, and that their performance is not adversely affected by changes in demand skewness patterns or daily order volume. On the other hand, he notes that the performance of sequential zone-based picking with batch deteriorates as order volume increases. Jane (2000) considered a sequential zone picking system, which he refers to as a relay picking system. He addressed the problem of assigning *n* products into *m* storage zones (one picker per zone) with the objective of minimizing the differences that might exist between each picker's total numbers of picks. Jane and Laih (2005) proposed a clustering algorithm for item assignment in a simultaneous zone picking system. They propose a similarity measure between any two items for measuring the co-appearance of both items in the same order. Accordingly, items frequently ordered together are located in different zones to minimize the idle time in the simultaneous zone systems. Le-Duc and De Koster (2005a) studied the same pick-to-belt systems. They extended their cost modeling analysis to a forward picking area including packing. This system is usually called a pick and pack system. They developed probabilistic MIP optimization model determining the zone size of a picker. The objective function of the optimization model is the overall time to complete a batch. It consists of four time components: travel time, set-up time, picking time, and correction time. Meller and Parikh (2006) focused on the problem of selection between a batch picking and a zone picking strategy. For this problem, they proposed a cost model to estimate the cost of each type of picking strategy. In their cost model, they considered the effects of pick-rate, picker blocking, workload-imbalance, and the sorting system requirement.

If one picker is assigned to more than one zone, there is sequencing problem of zone-visitation for a picker. Ho and Chien (2006) studied that a picker visits more than one zone to pick all the items in an order. They assume that no more than one picker can simultaneously be in the same zone. Then they determine the best zone-visitation sequence for a picker.

2.4.4 Comparative study for factors affecting on the performance of order picking operation

There are several factors that greatly affect the performance and efficiency of the pick operation. Major factors include the demand pattern of the items, the configuration of the warehouse, the slotting location of SKUs in the warehouse, the order batching method and the routing method used by the pickers to determine the sequence of the items to be picked. A variety of papers have been focused on the order picking performance. It is however difficult to find general conclusions since the performance depends heavily on the factors above mentioned. A comprehensive study that considers all the above factors has not been published at this time. A few results have been published where two factors are studied jointly.

De Koster et al. (1999) evaluated order batching and routing algorithms together, and Rubin and Jacobs (1999) studied order batching algorithms with different slotting policies. There are several studies on evaluating routing algorithms with different slotting policies. Petersen (1997) evaluated various routing heuristics and an optimal routine in a volume-based and random storage environment, comparing the performance of volume-based storage to random storage and examining the impact of travel speed and picking rates on routing and storage policy performance. The experimented results show the solution gap between routing heuristics and optimal routing is highly dependent on the travel speed and picking rate, the storage policy, and the size of the pick list. In addition, volume-based storage produced significant savings over random storage. Caron et al. (1998) developed a random and COI based slotting using ABC curve for assigning items to locations, and then developed analytical models for the expected travel distance of return and traversal picking policies required to pick the orders. In general, for COI-based storage systems, the return policy outperforms the traversal policy only for a low number of average picks per aisle and for skewed COI-based ABC curves. Hwang et al. (2004) evaluated the performance of three routing policies in the order picking policies (i.e. return, traversal, and midpoint policy) and compared the results of their analytical model with the results of simulation model developed. It is assumed that items are assigned to storage locations on the basis of the cube-per-order index (COI) rule in a low-level picker-to-part warehousing system. It is observed that for very small order size the return policy shows better performance, while for very large order size traversal policy performs better. In general, midpoint policy outperforms the other two. It indicates order picking heuristic performance in COI based slotting is similar to the random slotting (Hall, 1993). Le-Duc and De Koster (2005b, c) proposed a travel distance model for estimating the average tour length in 2-block warehouse when either S-shape or return method is used. The numerical results show that the return method is only better than S-shape for relatively small pick-list size and for very skewed storage assignments (ABC curves). This is similar to the finding in Caron et al. (1998) for the COI-based storage assignment.

2.4.5 Packing

Packing usually proceeds after the order picking operation and the consolidation operation. However, in the target zone-based carton picking system, the packing process should be performed during the order picking operation and the cartons after the order picking operation directly ship to customers. In this study, the packing operation (i.e., grouping line-items within an order into cartons with a limited capacity) is called *cartonization*. The planning of the cartonization should be finished before the order picking operation starts. It is necessary in practice to obtain the potential savings of order picking travel time by grouping line-items that are located in near slots, if order size is larger than carton capacity (shipping unit). The simplest way to reduce order picking travel time is to minimize the number of cartons by reducing the potential number of carton visit set-up time and travel time within zone by sharing a picking tour. There are a variety of traditional packing algorithms performed in the previous studies. In this section, we classified the several popular bin-packing algorithms by the compact of packing.

The description of the classical (general) Bin Packing (BP) problem is defined as follows: Given a finite set of $U = \{u_1, u_2, \ldots, u_n\}$ items and a rational size $s(u) \in [0,1]$ for each item $u \in U$, find a partition of U into disjoint subsets $U_1, U_2, \ldots, U_k$ such that the sum of the sizes of the items in each U_i is no more than 1 and such that k is as small as possible. Thus we can view each subset U_i as specifying a set of items to be placed in a single unit-capacity "bin", with our objective being to pack the items from U in as few such bins as possible. BP is polynomially equivalent to 3-PARTITION (BP $\propto$ 3-PARTITION). Because 3-PARTITION problem is well-known NP-complete class problem, we can say that BP is also NP-complete class problem. Since BP has "threshold existence" analog from the standard formulation such that "Is there a partition of U into disjoint sets $U_1, U_2, \ldots, U_K$ such that the sum of the sizes of the items in each U_i is B or less?". Thus, BP can be transformed into the optimization problem such that "minimizing the number of equal capacity bins necessary for the placement of a fixed set of pieces". Therefore, BP is NP-hard (Garey and Johnson, 1979).

1) Next fit algorithm (NF)

The simplest algorithm for the classical one-dimensional bin packing problem is Next Fit (NF). The algorithm first described by Johnson (1973). NF algorithm is described as follows: The next item is removed from the sorted list and tries to fit it onto the current bin. If the item fits, it is added and the process continues; otherwise the current bin is deemed full and closed and never reconsidered. A new empty bin becomes the current bin and the process continues until there is no item to be packed.

2) First fit algorithm (FF)

NF removes the next item from the sorted list and tries to fit it on a bin, but here is enhancement: FF tries the item on each partially-loaded bin, in order, and puts it on the first bin on which it fits. If it does not fit on any open bin, FF opens a new empty bin, put the item there and continues until there is no item to be packed. FF's worst-case behavior improves dramatically as the size of the largest item decline. Moreover, it maintains its advantage over NF in a certain situation.

3) Best fit (BF), worst fit (WF), and almost any fit algorithm (AAF)

The most famous of these rules is Best Fit (BF) algorithm. BF is similar to FF, but, BF tries the item on each partially-loaded shelf, in order, and puts it on the best bin on which it fits. BF seems better in principle than FF but has same worst case performance. Moreover, it is not observed to perform any better on average case analysis. FF packing rule can be implemented to run in time $O(n \log n)$. BF and FF can provide much different packing for individual lists. Nevertheless, all the performance results in worst case for FF hold for the performance results for BF as well (Johnson 1973, Johnson et al. 1974, Johnson 1974).

There are plausible packing rules for which the results of FF and BF are not able to hold in worst fit (WF). Consider the algorithm WF, in which each item a_i is packed in the partially-filled bin with the lowest level, assuming it fits, and otherwise starts in a new bin. Worst case performance ratio of WF and NF is same so that WF gets no value out of the fact that it never closes a bin. It takes only a slight modification to this algorithm to dramatically improve it. Let us say that an online bin packing algorithm is Any Fist (AF) algorithm if it never starts a new bin unless the item to be packed does not fit in any partially-filled bin in the current packing, and that it is in addition Almost Any Fit (AAF) algorithm if it never packs an items into a partially-filled bin with the lowest level unless there is more than one such bin or that bin is the only one that has enough room.

4) First fit decreasing (FFD) and best fit decreasing (BFD) algorithm

There are dangers in lists of items sorted by increasing size. Thus a natural idea for improving on FF once the online restriction is removed would be to sort the list in some other way before applying the First Fit packing rule. In the First Fit Decreasing (FFD) algorithm, the items are first sorted in order of non-increasing order size, and then the FF packing rule is applied. The algorithm best Fit Decreasing (BFD) is defined analogously, using the BF packing rule. The performance of FFD and DFD over FF and BF is dramatically improved (Johnson, 1973).

2.5 Summary

In this chapter, we surveyed the literature on warehouse operations. The warehouse operations are classified into four main areas: receiving, storing, order picking, and shipping. We mainly focused on the slotting operation and the order picking operation in this dissertation. There are two decision problems in the zone based carton picking systems. First one is *slotting* problem which determines an assignment of SKUs to slots. The other one is *cartonization* problem which determines a grouping of line-items within an order to cartons with a limited capacity.

In chapter 3, we propose a MIP model and heuristic models on the *slotting* problem given specific information of slotting of SKUs for the zone-based carton picking systems. When the number of picking items per picking tour is increased, we need more efficient slotting method to reduce order picking cost. The correlated slotting using the correlation between SKUs is one of the efficient slotting methods to minimize order picking cost in large number of items per picking tour. In chapter 3, we propose a MIP model and a simulated annealing improvement heuristic method using the correlation between SKUs based on a COI–based initial slotting solution under specific information of the cartons in a pick wave for the zone-based carton picking systems. In zone-based order picking systems, there are a few assignment problems about SKUs to zone or picker to zone (Jane 2000, Jane and Laih 2005, and De Koster and Yu 2008) and about an analytical modeling for expected picking time (DeKoster, 1994 and Yu and DeKoster, 2008, 2009). However, we have found no research for the slotting problem finding SKUs to specific slot locations using specific pick wave information in the zone-based carton picking systems.

In chapter 4, we propose a MIP model and a heuristic on the *cartonization* problem given specific information of slotting of SKUs for the zone-based carton picking systems. The cartonization essentially has the same characterization with order batching in the fact that it groups line-items into a carton to reduce the order picking cost by sharing a picking tour with the line-items being located in near slots. Several papers are found developing MIP model for the order batching. Most of the papers deal with the order picking systems with a specific front and back cross aisles, a P/D point being located in the left or center-point in the front cross-aisle, and a objective function to maximize the proximity between orders or minimize the sum of batch traveling distance (Chen and Wu 2005, Hwang and Kim 2005, Bozer and Kile 2008). However, there is no study to formulate mathematical model on catonization in zone-based carton picking systems. In heuristics, a variety of the order-batching heuristics have been studied in a number of specific order-picking systems. Since the order batching has essentially same characterization with the cartonization by sharing a picking tour with line-items being located in near slots, we searched and classified a variety of order batching papers for finding whether the grouping methodologies in the order batching can be applied in cartonization. As we mentioned above, the cartonization in this dissertation is different from the order batching in that the cartonization grouped line-items within an order into cartons being different from grouping orders in the order batching. As far as we know, there have been no literatures directly related to the catonization in zone-based carton picking systems. The cartonization is necessary in practice to obtain the potential savings of order picking travel time, if the size of an order is over the carton capacity or even less than the capacity. To solve the cartonization, we first considered the packing algorithms. A various traditional packing algorithms known as *bin-packing* have been studied. By applying the traditional packing algorithm into cartonization, it can obtain the potential picking time saving by sharing line-items with a picking-tour by reducing the number of cartons or carriers defining a picking-tour. The packing algorithms, however, minimize the number of cartons/carriers. Thus, it potentially provides a sub-optimal solution minimizing the order picking time or travel-distance in cartonization. In this chapter, we propose a new cartonization heuristic using a traditional bin-packing algorithm and geographical slotting information of SKUs adapting in order batching research.

The slotting operation in this dissertation is controlled in a more dynamic manner. In particular, different sets of SKUs are picked on short-term periodically and the entire picking area is periodically re-slotted, in the target environment the periods are typically quite short (e.g. one day). The decision for an efficient slotting depends on the decision for an efficient cartonization of a pick wave. Therefore, the decisions for the slotting and the cartonization must solve simultaneously. The slotting problem (or the cartonization) under the cartonization (or the slotting) being given, has been extensively studied. However, we have found no research to study both operations simultaneously under dynamic warehouse replenishment environment. To improve an additional performance, it is necessary to develop the two problems simultaneously in a dynamic whole warehouse replenishment environment. In chapter 5, we proposed an iterative slotting and cartonization heuristic using the slotting heuristic procedure in chapter 3 and cartonization heuristic procedure in chapter 4.

The literature on warehouse problem has been grown, because warehouse cost substantially increased (Council of Supply Chain Management Professionals, 2003-2008). Based on the literature review paper (Gu et al. 2007), more than 95% papers (120 papers / 124 papers) on warehouse operations are focused on slotting and order picking operation. Thus, the scope of the literature review in this chapter is confined the slotting operation and the order batching operation in order picking operation, because the targeting zone-based carton picking system in this dissertation is also closely related to both operations. We believe that this chapter enhances the understanding of the relation between two critical warehouse operations for the zone-based carton picking system and the difference of problem solving methods with the previous research.

Chapter 3
Slotting Method for Zone-based Carton Picking Systems

3.1 Introduction

The warehouse slotting problem involves determining an assignment of SKUs to picking slots to support order picking systems. Clearly a “good” slotting is one in which SKUs that are picked together into the same carton are also located near one another in the picking area. The traditional slotting problem uses long-term SKU demand correlations to identify a good slotting and re-slots the warehouse warranted when the SKU correlation structure is changed. The slotting operation in this study is based on a more dynamic environment. In particular, different sets of SKUs are picked on different days or short-term period and the entire picking area is re-slotted between each pick wave. As such, the long-term SKU demand correlations are of limited use and the specific correlations in a given pick wave can be exploited to identify good slotting for the specific pick wave. In this chapter, we address an efficient slotting method for zone-based carton picking systems under the dynamic replenishment environment described in Chapter 1. (i.e., entire warehouse is replenished with SKUs for a pick wave on the next short-term period).

The rest of Chapter 3 is organized as follows. Section 3.2 describes a mixed integer programming (MIP) slot assignment model for zone-based carton picking systems and a two-phase heuristic is developed to solve the dynamic slotting problem for large problems in Section 3.3. In the first phase, COI based slotting is performed. In the second phase, four types of improvement heuristics are developed to solve the dynamic slotting problem. In Section 3.4, the experimental parameters are presented. Three main results are reported in Section 3.5. First, a solution of the best heuristic model is compared with the optimal solution of MIP model. It shows that how the heuristic provides a good solution within a short computing time. Second, heuristic convergence test is presented. Last, the performance of four heuristics is presented in the large problems. It shows how the performance of the heuristics is affected by changing the experimental factors. Finally, Section 3.6 concludes the chapter with some promising research directions for further research.

3.2 MIP model for slotting problem

In this section, we introduce a MIP formulation to determine the slotting of SKUs in a carton picking system. The subscripts, parameters, and variables for the model are defined as follows:

J : number of cartons, $(j = 1, \ldots\ldots J)$

K : number of SKUs in forward picking area, $(k = 1, \ldots\ldots K)$

M : number of zones, $(l = 1, \ldots\ldots M)$

N : number of slots per aisle, $(m = 1, \ldots\ldots N)$

C_{jk} : indicator parameter set to 1 if SKU k is assigned to carton j, otherwise 0.

CS : carton setup time for pick-to-light loading due to a carton visiting to a zone.

S_m : setup time for zone m at the beginning or a pick wave.

P_n : picking time in slot n.

W_n : walking time to slot n.

F : maximum available picking time.

The decision variable set for this slotting model is x_{kmn}, which is equal to 1 if SKU k is assigned to slot n in zone m; and 0 otherwise. The remaining variables depend on the value of x_{kmn} and are defined as follows:

p_m : total completion time of a picker in zone m for processing cartons

t : pick wave makespan

d_{jm} : total walking time and carton setup time of carton j visiting zone m.

The completion time for cartons of a picker assigned in zone m is as follows ignoring starvation time as described in Section 1.1:

$$p_m = S_m + \sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{n=1}^{N} P_n C_{jk} x_{kmn} + \sum_{j=1}^{J} d_{jm}, \qquad \text{for } \forall m \qquad (3.1)$$

Then the 0-1 mixed integer formulation of dynamic slotting model (DS_MIP) is formulated as follows:

$$\text{(DS\_MIP):} \quad \min \quad t \tag{3.2}$$

subject to:

$$\sum_{m=1}^{M}\sum_{n=1}^{N} x_{kmn} = 1, \qquad \text{for } \forall k \tag{3.3}$$

$$\sum_{k=1}^{K} x_{kmn} \le 1, \qquad \text{for } \forall m, n \tag{3.4}$$

$$(CS + 2W_n)\sum_{k=1}^{K} C_{jk} x_{kmn} \le d_{jm}, \qquad \text{for } \forall j, m, n \tag{3.5}$$

$$p_m \le t, \qquad \text{for } \forall m \tag{3.6}$$

$$t \le F, \tag{3.7}$$

$$p_{m+1} \le p_m, \qquad \text{for } \forall m \setminus M \tag{3.8}$$

$$x_{kmn} \in \{0,1\}, \qquad \text{for } \forall k, m, n \tag{3.9}$$

$$d_{jm}, p_m, t \ge 0, \qquad \text{for } \forall j, m \tag{3.10}$$

Constraint set (3.3) ensures that each SKU is assigned exactly one slot. Constraint set (3.4) ensures that each slot contains at most one SKU. Constraint set (3.5) ensures that the total picking process time for a picker assigned in zone m for carton j is greater than the setup time for visit zone m for carton j plus the two times of the travel time to the slot assigned a SKU k in carton j from the zone initiation point. Constraint set (3.6) ensures that the pick time per picker in zone m is less than the pick wave makespan of pickers, t. Constraint (3.7) ensures that the variable t is less than the maximum available picking time. Constraint set (3.8) helps us to ignore the symmetry of solutions and reduces feasible solution search space. This constraint set forces the total picking processing time for picker in zone m to be greater than the time for picker in zone m+1. It eliminates alternative optimal solutions when zone size increases (Bozer and Kile, 2008). Constraint set (3.9) and (3.10) indicate that the decision variables are 0-1 integer and non-negative. The proposed MIP formulation provides an optimal solution. The size of formulation makes it difficult, if not impossible, to solve. This difficulty stems from the number of constraints and integer variables. This formulation has KMN binary variables and $K+2M+MN+JMN$ constraints. For example, the total number of variables and constraints from the target DC in a medium size of problem (Cartons: 300, SKUs: 540, Zones: 15) includes 296,116 variables and 167,611 constraints. Thus, CPLEX failed to find exact solutions before running out of memory in a number of cases. Also, this problem is known as NP-hard. If there is only one zone and each carton has only two line-items, this problem is equivalent to a well-known Quadratic assignment problem (QAP) (Frazelle, 1990). According to Garey and Johnson (1979), the QAP problem is strongly NP-hard, and then, our problem is strongly NP-hard, too. Thus, it is necessary to develop heuristic to solve the problem within a limited time constraint. However, the best feasible solutions obtained by CPLEX are useful to show the efficiency of the heuristic solution.

3.3 Heuristics for slotting problem

The MIP model for the slotting problem is NP-hard. Thus, an efficient heuristic method is needed to find a good solution among the large number of potential solutions. A proposed approach for such a situation is to use a search procedure. The search procedure attempts to explore a subset of the solution space in an attempt to identify a good solution. However, there is a trade-off between the computational efficiency and the solution quality. The basic search procedures have two phases such that a good initial solution is found by using a pick wave demand and then the initial solution is improved by changing it in some way. For initial slotting assignment, a slotting using cube-per-order index (COI) is proposed in that the most demanded SKUs are assigned into the "best" slots. The best slot means the nearest slot in time from a depot. SKUs are sorted in a descending order of quantity picked. Slots are sorted in increasing order of travel time to the picking location. In the special case of each carton having one line-item, the COI-based slotting represents the optimal solution. However, in the case of multiple line-items in a carton, the COI-based slotting cannot guarantee the optimal solution. For finding a better solution, the initial solution then is perturbed or altered in some way and the new solution is evaluated. If the new solution improves the objective, the change is kept and the new solution becomes the current solution. Otherwise, the change is discarded. The process repeats until a stopping criterion is satisfied. We propose four different types of improving search heuristics in this study. There are several issues that distinguish various search heuristics: how to perturb or alter the solution, which changed solutions to keep, and when to stop the search heuristic.

One common method for perturbing a solution is to perform pairwise interchange. In this method, two slots are selected and SKUs in the slots are exchanged. Other method for perturbing a solution is to perform correlated interchange. The method is based on the idea that items that appear together in the same carton should be located near each other in the picking area. Thus, the procedure first calculates the correlation between each pair of SKUs. Correlation is defined as the number of times that two SKUs are assigned in the same carton during a pick wave. The correlation list is then used to improve the base solution. The pair of SKUs with the highest correlation is selected and an interchange is made such that these SKUs are located next to each other in the rack face. If the interchange improves the solution, then it is kept. Otherwise, the interchange is not used. The procedure continues by considering the next pair of SKUs in the list. The correlated interchange is originally proposed by Smith and Peters (2001) and Smith and Kim (2008) examined the performance of correlated slotting method using correlated interchange by the various correlated carton lists. Pairwise interchange and correlated interchange methods are used as the basis for the second-phase of the heuristics in this study.

The second issue, which solutions to keep, also varies based on the heuristic method. Most of the search heuristics focused on keeping solutions which improve the objective, although these methods may become trapped in local optima. In local search, two types of acceptance rules to keep the improved solution are found. Common method is to accept any solution that improves the objective. The other method is to accept the solution that provides the best improvement in the objective among a set of improving solutions. Some heuristics probabilistically accept non-improving solutions in the objective hoping to expand search space. In these heuristic procedures, a global optimum can sometimes be found by escaping the local optima accepting the non-improved solutions. However, note that these procedures are not guaranteed to find the global optimal solution. In this study, we propose two types of local search heuristics and two types of global search heuristics.

Finally, the issue of when to stop the search must be addressed. This issue impacts the trade-off between computational time and solution quality in that the longer the search procedure is allowed to continue the more opportunity to improve the solution. Common approaches are to terminate when no further improvement is possible, when no improvement has been achieved for some predetermined number of solutions, when a specified number of solutions have been tried, and/or when a predetermined time limit is exceeded. For the comparison of heuristics in Section 3.5.2., the heuristics are terminated when no further solution is found for SA-C heuristic. For the comparison of heuristics for large size of problems in Section 3.5.3, the heuristics are terminated when no improvement has been achieved for some predetermined number of solutions and when a predetermined time limit is exceeded.

Based on the three issues for a search heuristic, we propose two local heuristics and two global heuristic using simulated annealing algorithms with pairwise interchange and correlated interchange for COI-based initial slotting. The detail algorithms are explained as following sections.

3.3.1 Steepest descent neighborhood slotting heuristic (SD)

The steepest descent neighborhood slotting improvement methodology (SD) uses pairwise interchanges for improving an initial solution in second-phase in this section. It evaluates all pairs of potential interchanges and chooses the solution with the most improved objective. It then reevaluates all pairs and continues this process until there is no improvement by interchanging solutions. Unfortunately, it is not guaranteed to reach the global optimal solution, and the starting solution obtained is important for solution quality for the final solution. Furthermore, it is time-consuming since it must evaluate the square of the total number of slots per iteration.

3.3.2 Correlated slotting heuristic (CS)

The correlated slotting improvement methodology (CS) developed attempts to exploit the problem using specific information about the cartonization. This procedure is based on the idea that items that appear together in the same carton should be located near to each other in the picking area. Thus, the procedure first calculates the correlation between each pair of SKUs. Correlation is defined as the number of items that two SKUs are assigned in the same carton during a pick wave. The correlation list is then used to improve the base solution. The pair of SKUs with the highest correlation is selected and an interchange is made such that these SKUs are located next to each other in the rack face. If the interchange improves the solution, then it is kept. Otherwise, the interchange is not used. The procedure continues by considering the next pair of SKUs in the list. The general steps of the slotting improvement procedure are as follows:

Step 1 Find an initial assignment based on the relative demand for particular SKUs and the relative preference of slot assignment of slots based on their proximity to the zone initiation point. This procedure is the traditional cube-per-order index (COI) based method.

Step 2 Calculate the correlation between each pair of SKUs and sort all pairs of SKUs in descending order. (The correlation list orders all pairs of SKUs in decreasing order of the number of times that the SKUs appear together in the same cartons. As such, this method attempts to iteratively move SKUs that appear together in the same carton closer to one another in the picking area.)

Step 3 Pick the pair of SKUs with the highest correlation and generate a new solution using a correlated interchange, in which the SKUs in the selected pair are located next to each other in the rack face. Update the correlated list by the pair of SKUs with the highest correlation as the pair of SKUs with the next highest correlation.

Step 4 Evaluate the new solution with the correlated interchange and compare the new solution with the best solution. If the new solution is better than the best solution, update the best solution and go back to Step 3. Otherwise, the correlated interchange is not to be used and go back to Step 3. If the solution is improved after the correlated list is consumed, then go back to Step 2. Otherwise STOP.

The correlated slotting (CS) provides better performance than the steepest decent heuristic (SD) in the large scale problems, because the CS quickly finds improved solutions by the correlated SKUs being slotted together, while the SD heuristic should search the entire neighbor-hood solution space to obtain an improved solution. However, the CS cannot escape a local optimal solution, once the solution falls in the local optimal solution. In order to escape the local optima, we propose simulated annealing slotting algorithms (SA) in the study in following section.

3.3.3 Simulated annealing slotting heuristic

Simulated annealing was first proposed by Kirkpatrick et al. (1983). SA is a technique developed to overcome some of the difficulties associated with the local optimum heuristic methods such as the steepest decent or the correlated slotting improvement heuristics mentioned in previous sections. SA differs in that the procedure uses random selection and will sometimes accept non-improving moves hoping to expand the search space and ultimately reach a better overall solution. The non-improving moves are probabilistically performed using Boltzman probability mass function as follows

(Wolsey 1998):

$$p(T) = \exp(-\Delta Z / T),$$

where T is the current temperature, $\Delta Z = Z(\mathbf{s}_c) - Z(\mathbf{s})$, and $Z(\mathbf{s}_c)$ and $Z(\mathbf{s})$ are the candidate and the current objective function value after and before interchange of SKUs, respectively.

SA algorithm was introduced as a heuristic approach to solve numerous combinatorial optimization problems. Burkard and Rendl (1984) and Whilhelm and Ward (1987), Herague and Alfa (1990) and Meller and Bozer (1996) solved QAPs using SA. Burkard (2002) states that SA yields excellent performance in QAPs. We also choose SA to solve the slotting problem, because both problems essentially have a same decision; i.e., departments to locations and SKUs to slots even if the slotting problem in this study is the larger number of assignments than facility layout problem.

In this section, we proposed two types of SA algorithms using pairwise interchange and correlated interchange. Meller and Bozer (1996) reported 7% improvement comparing steepest decent algorithm to SA algorithm using pairwise interchange (SA-P) in 40 department facility layout problem. As the problem size (i.e., SKUs, cartons, and line-items per cartons) becomes larger, the SA algorithm using pairwise improvement takes quite a long time to find a good solution given limited running time. In this section, we propose a SA algorithm using correlated interchange (SA-C) by a correlated list from the carton assignment. The SA-C algorithm dramatically improves solution performance in the initial stage comparing to SA-P, as well as it provides a good solution without converging to local optima. The general annealing scheme in this study is similar to Wilhelm and Ward (1987) and Meller and Bozer (1996).

The detailed algorithm for the SA-P and SA-C heuristic are given by following notations. Let

$\mathbf{s}^0 / \mathbf{s} / \mathbf{s}^c$:	the initial/current/candidate slotting solution vector
$\mathbf{s}^*$:	the current best slotting solution vector, which corresponds to the lowest pick wave makespan slotting by the algorithm.
$Z(\mathbf{s}^0)/ Z(\mathbf{s})/ Z(\mathbf{s}^c)$:	the objective values of initial/current/candidate slotting vector.
$Z(\mathbf{s}^*)$:	the objective values of the current best slotting vector.
$Z_j(\mathbf{s})$:	the objective values of the j th accepted candidate slotting vector in an epoch.
$\overline{Z}_e$:	the mean objective function value of an epoch, i.e., $\overline{Z}_e = \sum_{j=1}^{e} Z_j(\mathbf{s})/ e$.
$\overline{Z}_{2e}$:	the overall mean objective function value accepted in both the previous epoch and current one.
α :	the temperature cooling rate, which controls how fast the algorithm is “cooled-down”.
t_0 :	the initial temperature.
T:	a set of annealing schedule temperatures $\{t_1, t_2, t_3, \ldots\}$, where $t_i = t_0(\alpha)^i$, for $\forall i > 1$.
e :	the epoch length-fixed number of candidate solutions within each temperature.
ε_i :	the threshold value used to determine whether the system is in equilibrium. at temperature i.

$C(i, j)$:	the number of times SKUs *i* and *j* appear in the same carton for the given pick wave.
$L(k)$:	a SKU pair with *k*th high-ranked correlation of SKUs *i* and *j*, $C(i, j)$. where, $C(i, j) > 0$ is sorted in decreasing order of $C(i, j)$.
N :	the maximum number of successive temperature setting which do not produce a new $\mathbf{s}^*$.
T :	the termination time.
K :	the number of correlated list from $L(k)$, $k = 1, \ldots, K$

The parameters $\alpha, \beta, t_0, e, \varepsilon_i, M$ and N are specified *a priori*. Using the above notation, the detailed two SA heuristics are presented as follows.

In SA-P, we swap two SKUs in randomly selected slots. If the number of SKUs is less than the total number of slots, the swapping may move a SKU into empty slot. SA-C uses the information of correlated list and performs correlated interchange. This procedure is based on the idea that items that appear together in the same carton should be located near each other in the picking area. The SA-C improves the solution more quickly than SA-P in the initial stage. Therefore, The SA-C expects better performance than SA-P in large size problem within a given time limit. The general steps of the slotting improvement procedure for SA-P and SA-C are as follows:

Step 1 Generate an initial slotting vector $\mathbf{s}^0$ using COI slotting method. Based on the carton assignment of a given pick wave, generate the correlated SKU pairs $L(k)$, $k = 1, \ldots, K$.

Step 2 Set $\mathbf{s} = \mathbf{s}^0$. Given $\mathbf{s}^0$, compute the initial pick wave makespan, $Z(\mathbf{s}^0)$, and set $t_0 = Z(\mathbf{s}^0)/\tau$, $t_1 = \alpha t_0$, $i = 1$ and $m = 1$

Step 2a-SA-P Randomly select two SKUs and swap the SKUs. Store the resulting slotting vector (i.e. the candidate vector) as $\mathbf{s}^c$.

Step 2a-SA-C Select a random variable $k \sim U(1, K)$ and perform the correlated interchange using $L(k)$. Store the resulting slotting vector (i.e. the candidate vector) as $\mathbf{s}^c$ and increase m by 1.

Step 2b Compute decrease in pick wave makespan, $\Delta Z = Z(\mathbf{s}) - Z(\mathbf{s}^c)$. If $\Delta Z > 0$, go to Step 2d; otherwise go to Step 2c.

Step 2c Select a random variable $r \sim U(0,1)$. If $r < \exp(\Delta Z / t_i)$, go to Step 2d, otherwise go to Step 2a-SA-P or go to Step 2a-SA-C.

Step 2d Accept the candidate slotting solution vector $\mathbf{s}^c$ and current pick wave makespan; i.e., set $\mathbf{s} = \mathbf{s}^c$ and $Z(\mathbf{s}) = Z(\mathbf{s}^c)$. If $Z(\mathbf{s}) < Z(\mathbf{s}^*)$, then update the "current best" slotting solution vector and pick wave makespan; i.e., set $\mathbf{s} = \mathbf{s}^*$ and $Z(\mathbf{s}) = Z(\mathbf{s}^*)$. If e candidate slotting vectors have been accepted, go to Step 3; otherwise, go to Step 2a-SA-P or go to Step 2a-SA-C.

Step 3 If equilibrium has not been reached at temperature t_i; i.e., if $\left|\overline{Z}_e - \overline{Z}_{2e}\right| / \overline{Z}_{2e} \geq \varepsilon_i$, reset the counter for accepted candidate solutions and go to Step 2a; otherwise, set the number of epochs as 0, $i = i + 1$ and $t_i = t_0(\alpha)^i$. If $i < N$, go to Step 4; otherwise, STOP.

Step 4 If the running time is less than T, increase the number of epochs by 1 and go to Step 2a; otherwise STOP.

3.4 Experimental parameters

Four heuristics were coded in C++ and tested on several problems based on the experimental factors with several levels. Two types of parameters (i.e., system and operating parameters and SA parameters) are chosen before experimental testing. The system and operating parameters are chosen by the order picking system structure and operations of the order pickers. The parameters are referred by the technical report (Smith and Peters, 2001) which is studied on the case study of JC Penney distribution center in Plano, TX. The SA parameter values used for the experiment are: τ= 100, e=50, ε_i=0.01, N=10, T=10800, and α=0.997. The SA parameters were chosen based on preliminary experiments. Table 3.1 illustrates the order picking system parameters in this study. Each zone includes a rack with 54 slots (3 rows and 18 columns, and the slots are indexed as (current rack column – 1) $\times$ rack rows + current rack row. Walking and picking time are constant without acceleration. Since the picker picks items in different levels, the picking time is different by the level. Thus, the picking time weight are included based on the level. Two kinds of setup times (i.e., carton setup time and zone setup time) are considered.

Table 3.1 Order picking system parameters

System parameters		Operational parameters	
Rack rows (levels)	3	Bottom level weight	1.20
Rack columns	18	Middle level weight	1.00
Num. slots per zone	54	Top level weight	1.05
Unit walking time	1.4 secs/column	Carton setup time	10.80 secs/carton
Unit picking time	2.9 secs/SKU	Zone setup time	43.00 secs/zone

The heuristic experiments need to observe how the heuristics are affected by changing the level of factors for large problems. Since the performance heuristic algorithms depend on the number of SKUs (S), the number of cartons (C), the number of line-items per carton (LI), degree of correlation (CR), and the types of objective (O). We control these five parameters to several levels. To set up the experiment tests of the performances of heuristic for large problems in Section 3.5.3, we consider the following factors and the levels of the corresponding factors presented in Table 3.2. Since we could not have a real carton list data in this study, we have to generate a random carton list. In order to include the correlation between SKUs in the random carton list, we propose an effective correlated random carton list generation methodology with SKUs correlation. Based on the experimental factors, the generation of carton list is explained in detail in Appendix A. Using three randomly generated correlated carton lists, the above factors and their levels results in $2\times3\times3\times3\times2\times3$ = 324 instances of four heuristics, respectively. The running time was limited by three hours.

Table 3.2 Experimental factors

Factors	Levels
Number of SKUs (S)/ Number of zones$(S/54)$	540/10, 1080/20
Number of cartons (C)	300, 500, 700
Number of line-items per carton (LI)	10, 15, 20
Degree of correlation $(w=\{1,2,30\},\ n_i=0.1S)$	Low(w=1), Medium(w=2), High(w=30)
Types of objective	Pick wave makespan, Total cartons completion time

3.5 Experimental results

The MIP solution for slotting problem is executed by CPLEX 10.2. Because of the complexity of the problem, the MIP solution is difficult, if not impossible, to find an optimal solution within limited time in large size problem. We first compare SA-C heuristic with the optimal solution by using MIP model in small problems. For larger problems, the solution improvement between several heuristics is compared as computing time increases and then we test solution efficiency of the several heuristics by changing the experimental factors. The slotting heuristics are developed using $C++$ with Pentium IV 2.0 GHz CPU with 2.0 GB memory.

3.5.1 MIP and heuristic model comparison test

In the case of small size problems, we can compare MIP and SA-C heuristic. The factors on this test are number of zones(z), total number of SKUs stored (s), and number of cartons (c). The CPU time in MIP model is limited by 10 hours. The CPU time in the SA-C heuristic model is less than 60 seconds in the all the cases. In z, 2, 3, 4, and 5 zones are considered. Assuming that an even number of SKUs is stored in each zone, 3, 4, and 5 SKUs per zone are tested so that the total number of SKUs is $3z, 4z,$ and $5z$, (i.e., the total number of SKUs stored is 15, 20, and 25 in 5 zones). In c, 10, 20, and 30 cartons are considered. We fixed the average number of line-items in a carton and the degree of correlation is fixed as 5 and high $(w = 30)$. Therefore a total of 4 $\times 3 \times 3$ = 36 problem cases are tested. In each problem case, 10 instances are generated. Table 3.4 illustrates a summary of the average pick wave makespan of MIP solution and SA-C heuristic solution, the average relative deviation percentages between the makespan of MIP and SA-C, and the CPU time of finding a MIP solution. The average relative deviation percentages between the makespan of MIP and SA-C, Δ_h is defined as follows:

$$\Delta_h = \frac{1}{10}\sum_{i=1}^{10}\left(\overline{Z}_i^h - Z_i^{opt}\right)\times 100 \Big/ Z_i^{opt} .$$

where, $\overline{Z}_i^h$ be the average pick wave makespan of i th instance founded by using SA-C heuristic and Z_{OPT} be the pick wave makespan founded by using MIP model. We replicate the heuristic solution by 10 times to obtain a stationary solution. Therefore, $\overline{Z}_i^h = \sum_{i=1}^{10} Z_i^h / 10$.

Some problem instances remained unsolved even if we allowed 10 hours of CPU time. For calculating the average percentage deviation between the makespan of MIP and SA-C, we only considered the problems for which an optimal solution is found within the

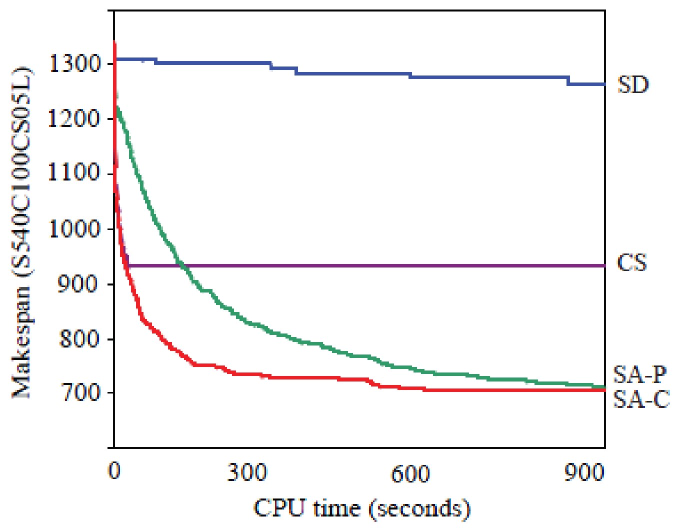
CPU time limit. If more than 50% of instances of MIP solutions (more than 5 instances) were not able to solve within 10 hours, we concluded that the MIP solution is failed. NA in table 3.4 indicates the problem cases that are failed. Overall, the average percentage deviation shows less than 5% from the optimal pick wave makespan even if the CPU time for finding an optimal solution using the MIP model exponentially increases (i.e., the number of SKUs stored in the zone and the number of cartons increase).

Table 3.3 Solution comparison between MIP and SA-C (APD: Average percent deviation, NOSF: Number of optimal solution found in MIP model)

Problems	MIP	SA-C	APD	NOSF	MIP time
Z2S06C10	256.16	256.16	0.00	10	0.07
Z2S08C10	265.27	265.61	0.15	10	0.78
Z2S10C10	263.45	267.07	1.39	10	17.17
Z3S09C10	224.12	224.97	0.41	10	3.28
Z3S12C10	223.03	227.26	1.95	10	393.10
Z3S15C10	216.88	223.74	3.22	10	3786.99
Z4S12C10	201.26	203.26	1.07	10	60.80
Z4S16C10	195.31	197.13	3.65	7	1372.39
Z4S20C10	193.68	201.45	2.89	5	5620.40
Z5S15C10	181.18	186.05	2.72	10	96.61
Z5S20C10	189.80	194.75	2.70	10	3221.63
Z5S25C10	172.61	177.63	3.15	5	9117.66
Z2S06C20	466.03	466.03	0.00	10	0.33
Z2S08C20	487.83	489.57	0.47	10	1.45
Z2S10C20	478.84	486.47	1.66	10	26.79
Z3S09C20	410.45	413.19	0.66	10	10.88
Z3S12C20	399.12	403.22	1.04	10	590.19
Z3S15C20	380.74	395.69	3.34	10	7023.91
Z4S12C20	363.60	369.23	1.86	10	699.61
Z4S16C20	326.95	342.45	4.12	8	13638.37
Z4S20C20	NA	NA	NA	1	NA
Z5S15C20	337.63	332.56	0.46	8	1354.38
Z5S20C20	312.93	320.69	3.87	6	20722.90
Z5S25C20	NA	NA	NA	1	NA
Z2S06C30	682.86	683.18	0.05	10	0.22
Z2S08C30	702.41	703.62	0.23	10	2.06
Z2S10C30	698.65	702.74	0.60	10	81.24
Z3S09C30	589.72	593.22	0.59	10	18.06
Z3S12C30	575.14	581.43	1.13	10	530.08
Z3S15C30	529.15	564.61	4.24	6	22929.65
Z4S12C30	517.20	522.85	1.15	10	494.11
Z4S16C30	477.10	490.37	3.53	5	26342.48
Z4S20C30	NA	NA	NA	1	NA
Z5S15C30	448.61	454.92	1.49	10	8069.86
Z5S20C30	447.10	452.98	3.44	5	35908.00
Z5S25C30	NA	NA	NA	1	NA

3.5.2 Heuristic convergence test

In this section, we compared the four improvement search heuristics during the entire running time: steepest decent neighborhood slotting (SD), correlated slotting (CS), simulated annealing using pairwise interchange (SA-P), and simulated annealing using correlated interchange (SA-C). Figure 3.1 presents the makespan improvement of each heuristic. In this test, we tested the same problem in three heuristic methods and truncated right when the solution in SA-C is stable. Two problems are executed to show the correlation impact on heuristic performance by changing different experimental factors. Based on 540 SKUs and 100 cartons, we tested two cases (i.e., 5 average line-items with low correlation between SKUs and 15 average line-items with high correlation between SKUs). In both the graphs in Figure 3.1, the worst heuristic is steepest decent neighborhood search (SD) and the best heuristic is simulated annealing using correlated interchange (SA-C). It shows about 35~45% savings from SD search heuristic. The main reason that SD heuristic finds a poor solution is that it takes substantial time to improve solution within the limited computing time because the current problem has large neighborhood sets. The correlated slotting improvement heuristic shows relatively a good solution in larger numbers of line-items with high correlation between SKUs. The correlated slotting method (CR) decreases the solution improvement gap with SA-C method from 32% in 5 line-items with low correlation to 10% in 15 line-items with high correlation. It indicates that the heuristics using correlation of SKUs performed better in the problems under larger pick-density (the picking numbers per picking tour) and larger feasible solution space. It also implies that the correlation information between SKUs is critical to improve the slotting solution under the carton data with high correlation and large number of line-items in the carton. The SA-C is quickly improved and it gives the best solution among other heuristics. In 5 average line-items with low correlation, the makespan of 706.12 is obtained in 592 seconds in SA-C compared to 950 seconds in the SA-P. In 15 average line-items with high correlation, the makespan of 1973.69 is obtained in 1778 seconds in SA-C compared to 2132 seconds in the SA-P. Thus, the computing time of SA-C is reduced by about 17~38% in obtaining a reasonable solution than the computing time of SA-P. We expect that the difference between the computing times of finding a reasonable solution using SA-P and SA-C increases as the number of line-items in carton increases and correlation of SKUs is high, because SA-C can quickly improve the solution using more correlated list generated by larger line-item and higher correlation.



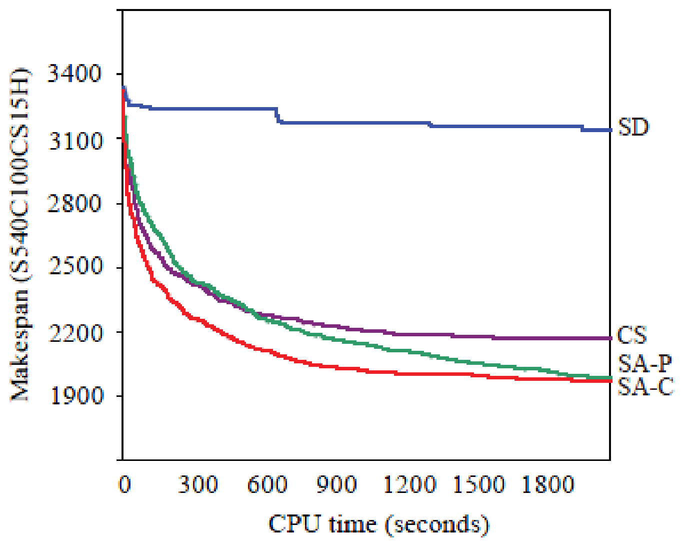


Figure 3.1 Heuristic solutions during computing time

3.5.3 Heuristic performance for large problems

To evaluate the performance of four different heuristic algorithms, we compare the pick wave makespan of each heuristic using a number of randomly generated problems. Since the complexity of the problem depends on the number of SKUs (S), the number of cartons (C), the number of line-items per cartons (LI), and the correlation of SKUs (CR), we control these four parameters to several levels. The levels of each control parameters are already shown in table 3.2. We randomly generated 3 problems and find an average pick wave makespan for each heuristic, respectively. The running time of the algorithms is fixed as 3 hours. Since SD and CR heuristic algorithms are local search algorithms, we added another termination condition, in which the algorithms stop when there is no improvement.

Table 3.4 shows the results of the pick wave makespan of each heuristic. The pick wave makespans of each of the four heuristics increase, as the number of SKUs, the number of cartons, and the number of line-items are large. In general, SA-C provides better pick wave makespan than other heuristics. In this table, the percent improvement between SD and SA-C in this table varies from 3.2% to 26.2% (the value of the most bottom right cells in SD and SA-C: 100 x (14089.5-13644.0)/14089.5 and the values of most upper left cells in SD and SA-C: 100 x (5354.4-7254.2)/5354.4). These values mean that 7.5 to 31.7 minutes of the working time savings of pickers can be obtained during one shift (8 hours) by the three hour slotting algorithm is performed. Since the running time is limited, the difference of the makespans among CR, SA-P, and SA-C becomes small as the number of SKUs becomes large and the number of line-items and the number of cartons become large. The heuristics using correlated interchange (CR and SA-C) has a relatively good performance compared to the heuristic without using correlated interchange (SD and SA-P). When the number of SKUs is 1080 (20 zone problem), we found five cases that CR (one of local search) performed better than SA-C in a 3 hour running time because SA-C takes considerable time to find a good solution due to a very large feasible solution space (i.e., the representation of the one solution in SA-C is a 1x1080 array). Thus, we performed additional tests for the five problem cases (i.e., S1080C700LI10H, S1080C700LI15H, S1080C700LI20H, S1080C700LI20L, and S1080C700LI10M) by increasing the running time from 3 to 6 hours. As we expected, the performance of the five the cases shows the makespan of the SA-C is 1.7~1.9 % better than CR.

The result of the percent improvement between COI initial solution and SA-C is presented in Table 3.5 and Table 3.6. Since the correlation list using the SA-C algorithm affects not only the performance of the COI solution but also the performance of the solution improvement using the SA-C, it is difficult to find a consistent trend in the performance of solution improvement by changing the degree of correlation in Table 3.5. Therefore, we present the average percent improvement of three degrees of correlations in Table 3.6. The cell in Table 3.6 is indicated by a set of a level of the number of SKUs, a level of the number of cartons, and a level of the number of line-items. The percent improvement between COI and SA-C is shown from 5.5% to 27%. The percent improvement becomes large as the number of line-item is small and the number of SKUs is small because the problem with small number of line-items and SKUs provides relatively smaller feasible solution space than the problem with large number of line-items and SKUs and it can obtain better solution more quickly under the 3 hours running time.

In the zone based picking systems, both the total carton processing time of the warehouse and the balance of the working time of pickers assigned to each zone are important to improve the productivity of the order picking process. Thus, we compared the performance of two objective functions (i.e, minimizing the pick wave makespan of pickers and minimizing the completion of total carton processing time). Table 3.7 summarizes the results of the percent improvement between SA-C under pick wave makespan objective function and SA-C under total carton completion time objective function. This table shows that there is a consistent increase in solution performance when the objective function is switched from pick wave makespan to the total carton completion time. The difference between pick wave makespan and total carton completion time increases, as the number of SKUs and the number of cartons increase.

Table 3.4

Average pick wave makespan for four heuristics

			SD			CR			SA-P			SA-C		
S	C	CR	LI = 10	LI = 15	LI = 20	LI = 10	LI = 15	LI = 20	LI =10	LI = 15	LI = 20	LI = 10	LI = 15	LI = 20
540	300	L	7254.2	9677.9	11467.0	5757.7	8108.6	10001.7	5550.3	7843.5	9746.5	5354.4	7678.0	9616.1
		M	7166.9	9545.2	11386.0	5679.7	8031.4	9922.4	5464.7	7800.2	9659.7	5282.5	7571.3	9479.4
		H	6352.3	8023.9	9258.8	5135.6	6881.4	8103.7	4946.4	6675.7	7722.2	4795.5	6597.6	7696.3
	500	L	12526.3	16661.7	19413.5	10676.9	14774.3	17841.8	10500.0	14351.5	17511.0	10255.2	14366.7	17499.2
		M	12248.6	16090.9	19167.6	10589.1	14499.8	17568.7	10347.1	14271.2	17291.9	10165.0	14114.4	17232.1
		H	10927.8	13504.7	15904.9	9301.6	12256.4	14444.6	9275.5	12059.3	14020.3	9122.3	12046.8	14001.8
	700	L	17926.3	23360.4	27385.7	15908.2	21573.4	25675.7	15633.5	21260.5	25477.0	15468.9	21090.7	25540.5
		M	17252.1	22840.2	27014.4	15436.0	21133.0	25272.1	15321.0	20944.8	25080.9	15067.5	20754.9	25103.0
		H	15338.3	19288.3	22654.3	13757.6	17865.7	20950.0	13562.9	17619.6	20415.0	13560.1	17714.0	20542.6
1080	300	L	4030.2	5766.1	7225.0	3279.6	5018.0	6564.7	3408.0	5056.4	6515.2	3083.8	4787.8	6307.4
		M	3962.9	5716.2	7277.6	3243.1	4976.6	6490.0	3379.3	5041.3	6432.2	3047.1	4797.6	6241.0
		H	3519.6	4695.9	5827.4	2886.1	4174.4	5106.9	2985.9	4200.4	5286.4	2754.0	4058.2	5011.0
	500	L	7013.8	9958.5	12483.5	6222.6	9185.9	11719.0	6335.6	9198.9	11706.3	6026.3	8990.3	11567.9
		M	6816.9	9726.0	12236.8	6087.4	9042.5	11476.9	6241.2	9143.4	11601.5	5982.9	8835.2	11420.1
		H	5996.6	8145.5	9850.4	5412.3	7551.5	9106.3	5512.6	7614.2	9287.1	5272.3	7499.8	9001.2
	700	L	9912.6	14219.0	17792.9	9132.5	13341.1	16817.6	9324.7	13436.1	17018.2	9051.5	13261.9	16869.4
		M	9876.0	13813.8	17502.6	9067.7	13107.7	16528.4	9229.1	13288.7	16792.4	9039.4	13078.9	16643.5
		H	8640.5	11568.1	14089.5	7934.8	10951.5	13449.5	8146.8	11000.3	13647.6	7954.1	11003.8	13644.0

Table 3.5
Percentage improvement between COI initial solution and SA-C heuristic

			COI			SA-C			% improvement		
S	C	CR	LI = 10	LI = 15	LI = 20	LI = 10	LI = 15	LI = 20	LI = 10	LI = 15	LI = 20
540	300	L	7437.2	9899.1	11567.3	5354.4	7678.0	9616.1	28.0	22.4	16.9
		M	7286.6	9631.6	11460.3	5282.5	7571.3	9479.4	27.5	21.4	17.3
		H	6428.7	8105.3	9397.5	4795.5	6597.6	7696.3	25.4	18.6	18.1
	500	L	12717.1	16804.9	19499.5	10255.2	14366.7	17499.2	19.4	14.5	10.3
		M	12417.0	16281.8	19265.5	10165.0	14114.4	17232.1	18.1	13.3	10.6
		H	11063.7	13639.9	16020.3	9122.3	12046.8	14001.8	17.5	11.7	12.6
	700	L	18233.4	23478.0	27656.7	15468.9	21090.7	25540.5	15.2	10.2	7.7
		M	17451.1	23066.2	27233.8	15067.5	20754.9	25103.0	13.7	10.0	7.8
		H	15636.4	19376.0	22893.1	13560.1	17714.0	20542.6	13.3	8.6	10.3
1080	300	L	4040.0	5831.4	7326.5	3083.8	4787.8	6307.4	23.7	17.9	13.9
		M	4069.7	5809.8	7328.3	3047.1	4797.6	6241.0	25.1	17.4	14.8
		H	3508.0	4774.4	5891.9	2754.0	4058.2	5011.0	21.5	15.0	11.6
	500	L	7084.4	10026.9	12615.5	6026.3	8990.3	11567.9	14.9	10.3	8.3
		M	6910.4	9838.0	12318.9	5982.9	8835.2	11420.1	13.4	10.2	7.3
		H	6050.1	8279.0	9939.9	5272.3	7499.8	9001.2	12.8	9.4	6.4
	700	L	9956.7	14347.0	17973.1	9051.5	13261.9	16869.4	9.1	7.6	6.1
		M	9970.7	13915.0	17635.7	9039.4	13078.9	16643.5	9.3	6.0	5.6
		H	8764.0	11690.6	14320.3	7954.1	11003.8	13644.0	9.2	5.9	4.7

Table 3.6

Average percentage improvement between COI initial solution and SA-C heuristic

		COI			SA-C			% improvement		
S	C	LI = 10	LI = 15	LI = 20	LI = 10	LI = 15	LI = 20	LI = 10	LI = 15	LI = 20
540	300	7050.8	9212.0	10808.4	5144.1	7282.3	8930.6	27.0	20.8	17.4
	500	12065.9	15575.5	18261.8	9847.5	13509.3	16244.4	18.3	13.2	11.2
	700	17107.0	21973.4	25927.9	14698.8	19853.2	23728.7	14.1	9.6	8.6
1080	300	3872.6	5471.9	6848.9	2961.6	4547.9	5853.1	23.4	16.8	13.4
	500	6681.6	9381.3	11624.8	5760.5	8441.8	10663.1	13.7	10.0	7.3
	700	9563.8	13317.5	16643.0	8681.7	12448.2	15719.0	9.2	6.5	5.5

Table 3.7
Comparison of the percentage improvement between SA-C under pick wave makespan objective function and SA-C under total carton completion time objective function

		% improvement of MS			% improvement of TC		
S	C	LI = 10	LI = 15	LI = 20	LI = 10	LI = 15	LI = 20
540	300	27.0	20.8	17.4	27.7	25.1	23.4
	500	18.3	13.2	11.2	21.8	19.7	18.1
	700	14.1	9.6	8.6	17.9	16.0	14.3
1080	300	23.4	16.8	13.4	32.3	28.0	25.9
	500	13.7	10.0	7.3	24.0	19.7	17.3
	700	9.2	6.5	5.5	18.3	14.7	14.3

3.6 Conclusions

The problem in this chapter is the slotting problem of zone-based carton picking order picking systems given cartonization. For solving the problem, a MIP programming model is introduced and solved by CPLEX 10.2. Since the problem is NP-hard and the size of a real problem is very large, we proposed four different heuristic algorithms: two local search based heuristics and two simulated annealing heuristics. Before we test large size problems, we compare the SA-C heuristic (the most sophisticated heuristic we proposed) with MIP solution in small size problems. The average relative percentage deviation between the makespan of MIP and SA-C provide less than 5% from the optimal pick wave makespan even if the CPU time for finding an optimal solution using the MIP model exponentially increases, as the number of SKUs stored in the zone and the number of cartons increase. For large size problems, we compared the performance of four heuristics in a 3 hour running time. From these results, we highly recommend that one should use the SA-C heuristic under the slotting problem with a limited slotting planning time and large size of solution space, because it quickly decreases solution without converging to local optima in the large size of problem solution space. SA-C heuristic uses the correlated list, which is the set of SKU pairs assigned in at least one carton and it dramatically improves solution in initial stage. The size of the correlated list and the correlation strength of the correlated SKU pairs affect both COI initial solution performance and SA-C improvement performance. The percent improvement between SD and SA-C in this table varies from 3.2% to 26.2%. These values mean 7.5 to 31.7 minutes of working time savings of pickers can be obtained during one shift (8hours) by the three hour slotting algorithm is performed.

In this study, we assume that the line-items per carton are given generated by the correlated random carton list generation method (Kim and Smith, 2008, Appendix A). The best slotting depends on how to assign orders to cartons given the number of orders in a pick wave (i.e., cartonization) and also the best cartonization depends on how to assign SKUs to slots (i.e., slotting). Clearly these two assignment problems affect one another. In the further study, we expect that a potential improvement can be obtained by considering the two interrelating assignment problems concurrently or systematically. Before we consider the interrelating problems, it is necessary to develop an efficient cartonization method of our order picking systems.

Chapter 4
Cartonization Method for Zone-based Carton Picking Systems

4.1 Introduction

Cartonization groups line-items within an order into cartons with a limited capacity. It is necessary in practice to obtain the potential savings of order picking travel time by grouping line-items that are located in near slots. The simplest way to reduce order picking travel time is to minimize the number of cartons by reducing the potential travel time within zone by sharing a picking tour. However, minimizing the number of cartons cannot guarantee to minimize carton set up time because more items are contained in a carton and the carton potentially visits more zones than the carton containing small items. Clearly a “good” cartonization is one in which SKUs in the same order that are assigned together into the same carton and are also slotted near one another in the picking area. In this chapter, we address an efficient cartonization method for zone-based carton picking system under dynamic replenishment environment described in Chapter 1. (i.e., entire warehouse is short-term periodically replenished with SKUs for a pick wave on the next period).

The rest of Chapter 4 is organized as follows. Section 4.2 describes a mixed integer programming (MIP) cartonization model for zone-based carton picking system. Since the problem is known as NP-hard in Section 4.2, we develops two types of cartonization heuristics, in which one class is heuristics without slotting information and the other class is heuristics with slotting information in Section 4.3. For the cartonization heuristics without slotting information, both carton capacity compared to mean SKU volume (the expected number of items per carton) and the ratio of carton capacity to mean order volume are critical issues to minimize pick wave makespan. In Section 4.4, we examine the cartonization heuristics by various experiment parameters and compare the performance of the proposed cartonization heuristics. Finally, we conclude the study with a summary and discuss some future research in Section 4.5.

4.2 MIP model for cartonization problem

In this section, we propose a mixed-integer programming (MIP) formulation to determine the grouping of orders into a carton given the slotting and a specific pick wave in a zone-based carton picking system. The general subscripts, parameters, variables are already explained in Section 3.2. Additional subscripts, parameters, and variables for the model are defined as follows:

Subscripts

I :	number of orders in a pick wave, $i = (1, \ldots, I)$.
$\mathbf{N_0}$:	set of non-negative integers, $\mathbf{N_0} = \{0, 1, 2 \ldots, N\}$.

Parameters

Q_{ik} :	required number of SKU *k* in order *i*.
X_{kmn} :	indicator parameter set to 1 if SKU *k* is assigned to slot *n* in zone *m*.
V_k :	unit volume of SKU *k* expressed in cubic feet.

V : carton capacity expressed in containable cubic feet in a carton.

Variables

q_{ijk} : number of SKU k in order i assigned to carton j.

u_{ij} : 1, if order i is assigned into carton j. 0, otherwise.

c_{ijk} : 1, if SKU k in order i is assigned into carton j. 0, otherwise.

The decision variable set for this cartonization model is q_{ijk}, which is the number of SKU k in order i assigned into carton j. The remaining variables depend on the value of q_{ijk}.

The completion time for cartons of a picker assigned in zone m is as follows ignoring starvation time as described in Section 1.1:

$$p_m = S_m + \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{n=1}^{N} P_n X_{kmn} q_{ijk} + \sum_{j=1}^{J} d_{jm}, \qquad \text{for } \forall m \tag{4.1}$$

Then the 0-1 mixed integer model for the cartonization (C_MIP) is formulated as follows:

$$\text{(C\_MIP): min } t \tag{4.2}$$

subject to:

$$\sum_{j=1}^{J} u_{ij} \geq 1, \qquad \text{for } \forall i \tag{4.3}$$

$$\sum_{i=1}^{I} u_{ij} \leq 1, \qquad \text{for } \forall j \tag{4.4}$$

$$\sum_{k=1}^{K} c_{ijk} \leq M u_{ij} \qquad \text{for } \forall i, j \tag{4.5}$$

$$\sum_{k=1}^{K} c_{ijk} \geq u_{ij}, \qquad \text{for } \forall i, j \tag{4.6}$$

$$\sum_{j=1}^{J} q_{ijk} = Q_{ik}, \qquad \text{for } \forall i, k \tag{4.7}$$

$$\sum_{i=1}^{I}\sum_{k=1}^{K} V_k q_{ijk} \leq V, \qquad \text{for } \forall j \tag{4.8}$$

$$q_{ijk} \leq Mc_{ijk}, \qquad \text{for } \forall i, j, k \tag{4.9}$$

$$(CS + 2W_n)\sum_{i=1}^{I}\sum_{k=1}^{K} c_{ijk} X_{kmn} \leq d_{jm}, \qquad \text{for} \forall j, m, n \tag{4.10}$$

$$p_m \leq t, \qquad \text{for } \forall m \tag{4.11}$$

$$p_{m+1} \leq p_m, \qquad \text{for } \forall m \setminus M \tag{4.12}$$

$$u_{ij,}\, c_{ijk} \in \{0,1\}, \qquad \text{for } \forall i, j, k \tag{4.13}$$

$$q_{ijk} \in \mathbf{N_0} \qquad \text{for } \forall i, j, k \tag{4.14}$$

$$d_{jm}, p_m, t \geq 0, \qquad \text{for } \forall j, m \tag{4.15}$$

Constraint set (4.3) ensures that an order must be assigned to at least one carton. Constraint set (4.4) ensures that all SKUs assigned to a carton belong to the same order. Constraint set (4.5) and (4.6) ensure couple u_{ij} and c_{ijk}. The constraint sets provide all the SKUs in a carton must belong to the same order. Constraint set (4.7) ensures that total number of units of a given SKU for an order must be equal to the required number of SKUs for that order. Constraint set (4.8) ensures that the total volume of line-items assigned to a carton from a certain order must be less than or equal to the carton capacity. Constraint set (4.9) ensures that two variables, u_{ij} and c_{ijk} are coupled. Constraint set (4.10) ensures that the total picking process time for a picker assigned in zone *m* for carton *j* is greater than or equal to the carton visiting set up time plus two times of the travel time to the slot assigned a SKU *k* in the carton *j*. Constraint set (4.11) ensures that the pick time per picker in zone *m* is less than pick wave makespan *t*. Constraint set (4.12) helps us to ignore the symmetry of solutions and reduces feasible solution search space. This constraint set forces the total picking processing time for picker in zone *m* to be greater than the time for picker in zone *m*+1. It eliminates alternative optimal solutions when zone size increases. Constraint set (4.13) and (4.14) indicate that the decision variables are 0-1 integer and non-negative integer. Constraint set (4.15) ensures the remaining variables are non-negative.

The number of binary variables, which is a key point to decide the difficulty of an integer programming problem, in this formulation is $IJ + IJK$ (0-1 integers) and IJK (non-negative integers). For the number of constraints, our formulation has $I+2J+2M+2IJ+IK+IJK+JMN-1$. If there is only one zone, one unit in the line-items in each orders and the processing time (picking and walking time of a picker) of each line-items is same, the problem is equivalent to a well-known bin packing problem. Thus, the cartonization problem is equivalent to a set of *I* bin packing problems, because it has *I* orders. According to Garey and Johnson (1979), the bin packing problem is strongly NP-hard and then, our problem is strongly NP-hard, too. Thus, it is necessary to develop heuristic to solve the problem within a limited time constraint.

4.2.1 MIP testing in small problems

Several small problems are solved with the MIP model on with Pentium IV, 2.8 GHz CPU using ILOG CPLEX 10.2. Before the MIP model is tested, we defined several testing parameters. The number of line-items in each order is fixed at 5. The unit of each line-item in an order is generated by the discrete uniform distribution, *DU*(1,10). The volumes of SKUs are generated by the continuous uniform distribution, *U*(0.025, 0.500) ft^3. The carton capacity is defined by the mean value of SKU unit multiplied by the mean value of volume of SKU.

There are three control factors to decide the complexity of the problem. The control factors are number of zones (z), total number of SKUs stored (s), and number of cartons (c). First, we should define the number of cartons to contain the line-items ordered enough. If the number of cartons is defined as an arbitrarily large value to contain the line-items fully assignable, an optimal solution is not able to be found within a given MIP processing time. If the number of cartons is defined too small, we obtain a local optimal solution for a given the number of cartons. To decide the number of cartons, we assume that each line-item individually requires a carton in the worst case. We thus define the number of cartons (c) as the number of orders multiplied by the mean number of line-items within an order. For example, the number of line-items per order is fixed as 5 and the number of orders can be one of two values (2 and 4) in this test. Then the number of cartons (c) can be one of two values (10 and 20). The number of zones (z) can be one of four values (2, 3, 4, and 5). Since we assigned the even number of

SKUs in each zone to balance the storing items throughout the zones, the number of SKUs per zone is 3, 4, and 5. We vary the total number of SKUs (s) from 6 to 25 (6, 8, and 10 SKUs in 2 zones, 9, 12, and 15 SKUs in 3 zones, 12, 16, and 20 SKUs in 4 zones, and 15, 20, and 25 SKUs in 5 zones). In each problem, 10 instances are generated. Therefore, total tested problems are 4 x 2 x 3 x 10 = 240. The CPLEX running time in MIP model is limited by 10 hours. Table 4.1 shows the results of the number of optimal solutions found, the average MIP objective value, and average running time of the optimal solutions found. During MIP tests, we found that some instances remained unsolved even if 10 hours of running time is consumed and some instances are stopped unsolved because of insufficient memory. For calculating average MIP solution and average CPLEX running time, we only considered the problems for which optimal solutions are found within the CPLEX running time limit.

Table 4.1 Mixed integer model test results (RT: Running time (seconds) for MIP model, NOP: Number of optimal solution found, NFE: the number of problems for which feasible solutions were found but failed to reach the optimal solutions within a pre-set running time limit, NFA: the number of problems that CPLEX running fails because of insufficient memory during branch and cut algorithm during the pre-set running time limit)

Problems	MIP	RT	NOP	NFE	NFA
Z02S06C10	193.9	0.6	10	0	0
Z02S08C10	185.1	26.6	10	0	0
Z02S10C10	206.3	17.6	8	2	0
Z03S09C10	170.0	1.0	10	0	0
Z03S12C10	152.2	0.4	10	0	0
Z03S15C10	164.6	0.4	10	0	0
Z04S12C10	156.5	0.5	10	0	0
Z04S16C10	148.1	0.1	10	0	0
Z04S20C10	144.4	4.3	10	0	0
Z05S15C10	138.7	0.3	10	0	0
Z05S20C10	128.4	0.3	10	0	0
Z05S25C10	126.5	0.1	10	0	0
Z02S06C20	NA	NA	3	5	2
Z02S08C20	315.9	819.7	5	2	3
Z02S10C20	NA	NA	2	2	6
Z03S09C20	284.4	2322.9	9	1	0
Z03S12C20	268.6	102.1	7	2	1
Z03S15C20	NA	NA	2	2	6
Z04S12C20	223.2	794.5	10	0	0
Z04S16C20	235.1	168.2	10	0	0
Z04S20C20	248.3	2208.6	9	0	1
Z05S15C20	215.7	116.1	10	0	0
Z05S20C20	210.9	609.7	10	0	0
Z05S25C20	200.1	1686.9	10	0	0

If more than 50% of instances of MIP solutions (more than 5 instances) could not solve within 10 hours, we concluded that the MIP solution has failed. NA in Table 4.1 indicates the problem cases that have failed. The values in the ‘NOP’ column indicate the number of problems that could be solved optimally. The ‘NFE’ column shows the number of instances for which feasible solutions were found but failed to reach the optimal solutions within a running time limit (10 hours). The values in the ‘NFA’ column indicates the number of instances that CPLEX running fails because of insufficient memory during branch and cut algorithm during the running time limit. In MIP test, we found that the running time exponentially increases and the number of optimal solutions decreases within running time limit of CPLEX, as the number of cartons increases. As shown in Table 4.1, ‘NFE’ or ‘NFA’ become larger, as the number of cartons increased. This shows that an increase of the number of cartons inreases the running time and computing memory required to solve the problem optimally. Therefore, we conclude that the proposed MIP cartonization model is impractical as the complexity of the problem increases. Therefore, it is necessary to develop an efficient heuristic to solve for larger number of cartons.

4.3 Heuristics for cartonization problem

In this section, we classified the cartonization heuristics into two types, in which one class is the heuristics without slotting information and the other class is the heuristics with slotting information. In general, carton density (the number of items per carton) and grouping of the corresponding items to be assigned together in a carton are critical factors to minimize pick wave makespan in cartonization. The cartonization heuristics without slotting information cannot guarantee a good solution, because the geographical slot locations of the items cannot be known. Therefore, we can only control the number of items per carton to reduce pick wave makespan. As increasing the number of items per carton, we can obtain a potential reduction of the pick wave makespan by sharing the items in a picking tour. In this case, the cartonization is similar in characteristic with a classical bin packing problem because the objectives on both problems are to minimize the number cartons by increasing items per carton. Therefore, we propose several cartonization heuristics without slotting information using classical bin packing heuristics.

For the heuristics with slotting information, both carton density (the number of items per carton) and grouping of items to be assigned together in a carton are critical factors to minimize pick wave makespan. In this case, we propose several cartonization heuristics with slotting information for considering both the number of items per cartons and geographical slot locations for the items.

4.3.1 Heuristics without slotting information

Johnson et al. (1974) examine next fit decreasing and first fit decreasing heuristic, which are two of the most famous heuristics and show excellent worst case performances. The heuristics show a good performance in the batch loading and scheduling problem (BLSP) with non-identical job sizes and no grouping concept (Uzsoy, 1994). BLSP has a similarity with cartonization without slotting information and scheduling in this study. Therefore, we propose the next fit decreasing and first fit decreasing bin packing heuristics as the heuristic without slotting information. We call two cartonization heuristics as next fit by volume decreasing (*NFVD*) and the first fit by volume decreasing (*FFVD*) and both algorithms make cartonization as follows:

Procedure: NFDV

Step 1: Sequence orders in a pick wave in FCFS order.

Step 2: If there are no remaining orders in the pick wave, go to Step 4.

For each order in pick wave, sort the line-items within an order in decreasing order of the unit-volume of them and select a line-item in the sequence of the sorted line-items.

Step 3: If the remaining carton capacity of the current carton is available, assign one unit of the line-item in the order into the current carton. Otherwise, close the current carton, open a new carton as the current carton, and assign one unit of the line-item in the order into the current carton.

Recalculate the remaining carton capacity of the current carton.

Repeat Step 3, until every unit of the line-item is assigned into the carton.

If all of the units of the line-item are assigned into the carton, move to the next line-item for assignment. Repeat Step 3.

If there are no remaining line-items in the order, move to the next order and go to Step 2.

Step 4: Assign the cartons to the order picking system in an arbitrary order.

Procedure: FFDV

Step 1: Sequence orders in a pick wave in FCFS order.

Step 2: If there are no remaining orders in the pick wave, go to Step 4.

For each order in pick wave, sort the line-items within an order in decreasing order of the unit-volume of them and select a line-item in the sequence of the sorted line-items.

Step 3: Find the first available carton from first carton to the current carton and assign one unit of the line-item in the order into the carton.

If there is no available carton from first carton to the current carton, open a new carton as the current carton and assign one unit of the line-item in the order into the current carton.

Recalculate the remaining carton capacity of the current carton.

Repeat Step 3, until every unit of the line-item is assigned into the carton.

If all of the units of the line-item are assigned into the carton, move to the next line-item for assignment. Repeat Step 3.

If there are no remaining line-items in the order, move to the next order and go to Step 2.

Step 4 Assign cartons to the order picking system in an arbitrary order.

4.3.2 Heuristics with slotting information

In the cartonization, the assignment of line-item slotted in a zone to a carton is critical to reduce the pick wave makespan, because both walking time and carton set up time for a carton visit to a zone affects the processing time of a picker in each zone. Therefore the critical issues in the cartonization with slotting information are

1) How many items in the same zone are assigned together into a carton?

2) How close the items in the same zone are assigned into a carton?

In the heuristic using slotting information, we split line-items within an order into zones and sort line-items within the zone in decreasing order of the proximity from a zone initiation point to the slot location of the line-items. Then, we sequence the picking zone in descending order of the number of line-items to be picked and select a zone to assign associated line-items to be picked in the zone to cartons in the sequence of the sorted zones. In each zone, the sorted line-items are sequentially assigned to the opened carton. Once the line-items are completed in a zone, there are two types of heuristics. The current carton is closed and a new carton is opened for the next available zone with zone separation and the current carton continues to assign line-items for next zone without zone separation. Using the same procedure, we perform the cartonization process to the last zone. The cartonization procedure repeats until the last order performed. We call this heuristic as bin-packing heuristic using proximity with zone separation. Two types of the bin-packing heuristic using proximity with zone separation are proposed in this study (i.e., the next fit proximity decreasing with zone separation (NFDP-Z) and the first fit proximity decreasing with zone separation (FFDP-Z). The cartonization heuristics using proximity with zone separation by carton are expected to be better performance than the cartonization heuristics without slotting information (NFDV, FFDV) by reducing pickers walking time and the reduction of the carton set up time. However, it clearly results in more cartons than it is necessary used. Hence, we also proposed two relaxed heuristics by eliminating the procedure that a new carton is opened whenever the first line-item in a new zone is considered in NFDP-Z or FFDP-Z heuristic. We call the heuristics with relaxation as next fit proximity decreasing without zone separation (NFDP-WZ) and the first fit proximity decreasing without zone separation (FFDP-WZ). The algorithms are described in detail as follows:

Procedure: NFDP-Z or NFDP-WZ

Step 1: Sequence orders in a pick wave in FCFS order.

Step 2: If there are no remaining orders in the pick wave, go to Step 6.

For each order in pick wave, split the line-items into zones being assigned, in which the corresponding SKUs are slotted.

Step 3: For each zone, sort the line-items in each zone in decreasing order of the proximity from the zone initiation point to the slot of the line-item assigned.

Sequence the picking zone in descending order of the number of line-items to be picked.

Step 4: Select a zone to be cartonization in the sequence of the sorted zones.

If the algorithm is **NFDP-Z**, close the current carton and open a new carton.

Step 5: If the remaining carton capacity is available, assign one unit of the line-item in the order into the current carton. Otherwise, close the current carton and open a new carton as the current carton, and assign one unit of the line-item in the order into the current carton.

Recalculate the remaining carton capacity of the current carton.

Repeat Step 5, until every unit of the line-item is assigned into the carton.

If there are no remaining line-items in the zone, go to Step 5.

If all of the units of the line-item are assigned into the carton, move to the next line-item for assignment. Repeat Step 4.

If there are no remaining line-items in the order, move to the next order and go to Step 2.

Step 6: Assign cartons to the order picking system in an arbitrary order.

Procedure: FFDP-Z or FFDP-WZ

Step 1: Sequence orders in a pick wave in FCFS order.

Step 2: If there are no remaining orders in the pick wave, go to Step 6.

For each order in pick wave, split the line-items into zones, in which the corresponding SKUs are slotted.

Step 3: For each zone, sort the line-items in each zone in decreasing order of the proximity from the zone initiation point to the slot of the line-item assigned.

Sequence the picking zone in descending order of the number of line-items to be picked.

Step 4: Select a zone to be cartonization in the sequence of the sorted zones.

If the algorithm is **FFDP-Z**, close every carton assigning line-items in the previous zone even if the remaining capacities of cartons are available to assign quantities of the line-items in current zone and create a new carton.

Step 5: Find the first available carton from first carton to the current carton and assign one unit of the line-item in the order into the carton.

If there is no available carton from first carton to the current carton, open a new carton as the current carton remaining the previous carton is opened and assign one unit of the line-item in the order into the current carton.

Recalculate the remaining carton capacity of the current carton.

Repeat Step 4, until every unit of the line-item is assigned into the carton.

If all of the units of the line-item are assigned into the carton, move to the next line-item for assignment. Repeat Step 4.

If there are no remaining line-items in the order, move to the next order and go to Step 2.

Step 6: Assign cartons to the order picking system in an arbitrary order.

4.4. Computational experiments

To evaluate the performance of heuristics, we mainly compare the average pick wave makespan and the number of cartons used using a randomly generated problems. For achieving practical pick wave results, we define the base parameters. The value of the number of orders per pick wave is fixed as 200. The value of the number of line-items per order is classified into three levels: 5, 10, and 15. The value of the volume of SKUs is generated by a uniform distribution with *U*(0.025, 0.475) ft^3. The value of the unit of line-items is generated by a discrete uniform distribution with *DU*(1, 9). The carton capacity is fixed to 4.25 ft^3. Based on the base parameters, the mean order volume is 12.5ft^3 (=10x0.25x5). Therefore it approximately 3 cartons per order (=12.5/4.25) are required. In this study, we assume that the slotting of SKUs is predetermined. We can produce the slotting information for two types of slotting methods (i.e., random slotting or COI slotting) using a randomly generated pick wave (a set of orders).

The order picking system parameters are followed by the parameters in Chapter 3. Each zone includes a rack with 54 slots (3 rows and 18 columns, and the slots are indexed as (current rack column - 1) $\times$ rack levels + current rack row). Walking time and picking time are constant without acceleration being considered. As having different picking levels, the picking time weights are included (Bottom: 1.20, Middle: 1.00, Bottom: 1.05). Two kinds of set up times, carton set up time and zone set up time are considered.

Since we expect that the solution performance depends on several factors, we set the factors as control parameters. Based on the base parameters and system parameters, we analyze the solution performance by changing controlling parameters based on the base parameters. The three control parameters in this study are presented (the capacity compared to mean SKU volume (the expected number line-items per carton), the ratio of the carton capacity to the mean order volume, and the slotting methods)). In each testing problem, 30 test problems that are randomly generated. All solution approaches have been coded in C++ and run on a 3.4 GHz Pentium 4 PC with 2.99 GB of memory and operating system of Windows XP. Under these conditions, the CPU time to get a solution from each heuristic algorithm is less than 5 seconds in the various experimental test sets.

Table 4.2 shows the result of performance for NFDV, NFDP-Z and NFDP-WZ under COI slotting and random slotting. We compare the mean pick wave makespan (Mean) and the number of cartons (NC) with different level of line-items per orders (LI). NFDP-Z heuristic gives the lowest average pick wave makespan by showing 4732.0 and 7670.5 under COI and random slotting, respectively. Even though NFDP-Z presents lower pick wave makespan than DFDV, it requires impractical number of cartons used by showing almost twice number of cartons than NFDV to satisfy the pick wave makespan.

Table 4.2 Heuristic performance for NFDV, NFDP-Z, and DFDP-WZ
(a) COI slotting

	NFDV		NFDP-Z		NFDP-WZ	
LI	Mean	NC	Mean	NC	Mean	NC
5	3448.7	423	2788.8	853	3508.3	370
10	6770.8	739	4834.3	1444	6279.5	709
15	9814.4	1071	6572.9	1903	8392.5	1057
Average	6678.0	744	4732.0	1400	6060.1	712

(b) Random slotting

	NFDV		NFDP-Z		NFDP-WZ	
LI	Mean	NC	Mean	NC	Mean	NC
5	5278.1	423	4314.9	850	5391.7	367
10	11228.1	759	7918.0	1445	10182.0	727
15	16913.4	1089	10778.5	1896	13824.2	1071
Average	11139.9	757	7670.5	1397	9799.3	722

To reduce the number of cartons in NFDP-Z, we propose DFDP-WZ by relaxing the constraint for zone separation in NFDP-Z. In this heuristic, we follow the general procedures in MFDP-Z except eliminating a procedure in MFDP-Z that a new carton is opened whenever the first line-item in a new zone. NFDP-WZ still shows better pick wave makespan than NFDV by showing the relative percent improvement of pick wave makespan from 9.3% (=100 x (6678.0 - 6060.1) / 6678.0) to 12.0% (=100 x (11139.9 - 9799.3) / 11139.9) in COI slotting and random slotting, respectively. Furthermore, NFDP-WZ shows the reduction of the number of cartons used from 4.3% (=100 x (744 - 712) / 744) to 4.6% (=100 x (757 - 722) / 757) in COI slotting and random slotting, respectively. Therefore, we select NFDP-WZ as a representative cartonization method with slotting information for further tests.

We present the results of performance between two cartonziation heuristic methods without slotting information (FFDV and NFDV) and two cartonization heuristic methods with slotting information (FFDP-WZ and NFDP-WZ) in Table 4.3. To compare four heuristics, we control two parameters, the mean number of line-items and the slotting methods. The mean number of line-items per order (LI) can be one of the five values (5, 10, 15, 20, and 25) and slotting method can be one of the two slotting methods (COI and random slotting).

Since we randomly generate 30 instances for the combination of the levels for the control parameters, we compare the mean pick wave makespan (Mean), the standard deviation of the pick wave makespan (SD), and the mean number of cartons (NC) between the heuristics. In general, NFDV and NFDP-WZ provide lower pick wave makespan than FFDV and FFDP-WZ, because FFDV and FFDP-WZ search all the previous available cartons. Therefore, FFDV and FFDP-WZ require additional carton set up time and picking and walking time for the SKUs in the additional zones compared to NFDV and NFDP-WZ.

The heuristics with slotting information (NFDP-WZ and FFDP-WZ) provide less pick wave makespan than the heuristics without slotting information (NFDV and FFDV), as the number of line-items per order increases. With 5 line-items per order (LI), NFDV shows the lowest pick wave makespan and NFDP-WZ heuristic shows the lowest mean pick wave makespan as the number of line-items increases. Furthermore, NFDP-WZ needs only from 41 to 43 more cartons to FFDV (the densest packing method among four testing heuristics) for the case with 25 line-items per order (LI). These differences indicate that NFDP-WZ heuristic only requires more cartons from 2.4% (= 43/1697x100) to 2.5% (= 41/1693x100) than FFDV. Each heuristic achieves the better pick wave makespan in COI slotting than random slotting, but cartonization in the random slotting shows better relative percent improvement between NFDV (cartonization without slotting information) and NFDP-WZ (cartonization with slotting information) than cartonization in the COI slotting. For example, the percent improvement between NFDV and NFDP-WZ in the number of line-items per order (LI) as 25 under COI slotting is 24.2% (100 x (15861.5 - 12028.8) / 15861.5) and the percent improvement between NFDV and NFDP-WZ in the number of line-items per order (LI) as 25 under random slotting is 30.3% (100 x (30385.5 - 21194.3) / 30385.5). The example indicates that the cartonization with the poor slotting (random slotting) results in higher percent improvement between NFDV and NFDP-WZ than sophisticate slotting (COI slotting), because the travel time reduction is already obtained by the sophisticated slotting (COI slotting) when cartonization methods are tested under COI slotting. Therefore, there is relatively a small impact on the performance by the cartonization methods.

Table 4.3 Four heuristic methods performance comparison

(a) COI slotting

	FFDV			NFDV			FFDP-WZ			NFDP-WZ		
LI	Mean	SD	NC	Mean	SD	NC	Mean	SD	NC	Mean	SD	NC
5	3751.7	105.5	420	3448.7	81.2	423	3749.8	134.5	420	3519.8	117.6	370
10	7663.3	233.6	726	6770.8	155.2	739	7344.5	234.9	729	6275.8	201.0	709
15	11333.4	327.2	1047	9814.4	326.5	1071	10342.9	465.9	1052	8429.6	279.1	1057
20	14906.5	379.7	1374	12817.3	274.5	1411	13193.0	584.9	1381	10324.5	300.0	1403
25	18355.8	493.9	1697	15861.5	434.0	1744	15794.1	845.4	1704	12028.8	324.0	1740

(b) Random slotting

	FFDV			NFDV			FFDP-WZ			NFDP-WZ		
LI	Mean	SD	NC	Mean	SD	NC	Mean	SD	NC	Mean	SD	NC
5	5709.0	266.3	419	5278.1	237.2	423	5807.5	392.7	420	5410.5	220.9	367
10	12716.4	867.9	744	11228.1	751.2	759	11675.3	731.3	748	10264.6	635.6	727
15	19508.8	1137.4	1063	16913.4	1127.5	1089	16997.3	1049.8	1069	14026.7	730.2	1070
20	26970.4	1956.7	1368	23463.7	2105.0	1406	21958.9	1597.6	1374	17789.3	1396.0	1396
25	34958.5	3298.6	1693	30385.5	2779.2	1739	27943.5	2505.6	1700	21194.3	1729.6	1734

Table 4.5 compares the performance between NFDV and NFDP-WZ heuristic algorithms by changing the carton capacity compared to mean SKU volume (the expected number line-items per carton) and the ratio of the mean volume per order to the carton capacity. In this study, the number of line-items per carton is a decision variable. However, we can indirectly control the expected number of line-items per carton by adjusting a carton capacity given a mean line-item (SKU) volume, because the NFDV and NFDP-WZ cartonization methods are the assignment methods using bin packing heuristic and the line-items in a carton must be contained as many as possible in a fixed carton capacity.

Since NFDV performs better in the case without slotting information than FFDV and NFDP-WZ performs better in the case with slotting information than NFDP-Z in Table 4.3, we compare only two representative cartonization methods in Table 4.5. To evaluate the performance between the heuristic algorithms, we compare the relative percent improvement and the relative difference of number of cartons between NFDV and NFDP-WZ. Since the performance of the heuristics depends on the expected number of line-items per carton and the ratio of the carton capacity to the mean order volume, we fixed the mean volume of a line-item (SKU) as 1.250 ft^3 (=5x0.25), which is generated by the mean unit of line-items per carton as 5 from *DU*(1,9) and then the mean volume of line-items as 0.25 from *U*(0.025, 0.475).

Hence we control other two parameters to several levels. First, the carton capacity (CP) can be one of the four values, 6.25 ft^3, 12.50 ft^3, 18.75 ft^3, and 25.00 ft^3. In each carton capacity, a carton can contain 5, 10, 15, and 20 line-items, respectively because the mean volume of line-item (SKU) is 1.250 ft^3 (=5x0.25). Second, the ratio of carton capacity to the mean order volume (RT) are 1:1, 1:2, 1:5, and 1:10. Then there are 16 pairs of (CP, RT): (6.25, 1:1), (6.25, 1:2),…, and (25.00, 1:10). For each pair out of these 16 pairs, we randomly generate 30 problems. Each of Table 4.5(a) and 4.5(b) consists of 16 cells according to the level of CP and RT and each cell represents a summary of the results for the 30 test problems that are randomly generated. Each cell contains four kinds of values, each of which represents the average relative percent improvement (PI) and the standard deviation of the relative percent improvement in the first column in the cell and the average relative difference of the number of cartons (DNC) and the standard deviation of the number of cartons in the second column of the cell. For example, the values 2.3 and 4.8 in the first column and 51, and 6 in the second column of upper left corner, in the cell (6.25, 1:1) in Table 4.5(a) are derived in Table 4.4. The average and standard deviation of the relative percent improvement in the cell (6.25, 1:1) in Table 4.5(a) are 1.3 and 4.7 and the average and standard deviation of the difference of the number of cartons are 51 and 6, respectively.

Table 4.4 Mean and standard deviation of % improvement (PI) of pick wave makespan and difference (DF) of number cartons used between NFDV and NFDP-WZ heuristics for 30 problems in cell (6.25, 1:1)

	NFDV		NFDP-WZ		Comparison	
Problems	MS	NC	MS	NC	PI	DF
1.0	5104.5	303.0	4510.6	238.0	11.6	65.0
2.0	4566.7	291.0	4741.6	240.0	(3.8)	51.0
3.0	4573.3	282.0	4798.5	235.0	(4.9)	47.0
4.0	4732.4	295.0	4113.2	247.0	13.1	48.0
5.0	4825.4	283.0	4502.0	225.0	6.7	58.0
6.0	4381.7	274.0	4261.2	226.0	2.8	48.0
7.0	4221.1	270.0	4167.7	235.0	1.3	35.0
8.0	4831.8	304.0	4522.9	252.0	6.4	52.0
9.0	4343.1	293.0	4124.0	242.0	5.0	51.0
10.0	4894.9	283.0	4662.5	232.0	4.7	51.0
11.0	4626.5	299.0	4734.3	247.0	(2.3)	52.0
12.0	4569.4	292.0	4673.7	237.0	(2.3)	55.0
13.0	4400.8	299.0	4584.4	244.0	(4.2)	55.0
14.0	4468.0	295.0	4737.8	236.0	(6.0)	59.0
15.0	4314.2	268.0	4409.1	219.0	(2.2)	49.0
16.0	4720.6	284.0	4663.0	229.0	1.2	55.0
17.0	4554.9	290.0	4532.1	249.0	0.5	41.0
18.0	4780.7	283.0	4599.5	228.0	3.8	55.0
19.0	4280.8	288.0	4299.8	232.0	(0.4)	56.0
20.0	4162.6	294.0	4370.8	237.0	(5.0)	57.0
21.0	4469.5	270.0	4336.6	226.0	3.0	44.0
22.0	4735.6	297.0	4760.1	251.0	(0.5)	46.0
23.0	4680.8	289.0	4521.5	246.0	3.4	43.0
24.0	4299.2	290.0	4371.1	238.0	(1.7)	52.0
25.0	5075.3	299.0	4915.3	249.0	3.2	50.0
26.0	4636.0	278.0	4369.3	232.0	5.8	46.0
27.0	4660.8	310.0	4632.7	250.0	0.6	60.0
28.0	4112.0	289.0	4097.1	235.0	0.4	54.0
29.0	4998.7	293.0	4792.2	239.0	4.1	54.0
30.0	4565.5	287.0	4773.0	233.0	(4.5)	54.0
				Mean	1.3	51.4
				SD	4.7	6.2

The relative percent improvement between NFDV and NFDP-WZ vary from -1.3 to 64.6 in Table 4.5(a) and 4.5(b). Since the practical order picking system has a picking time restriction for a pick wave, the 64.6 percent improvement seems to be a significant one. For example, if we have 8 hours picking time restriction per pick wave, the 64.6 improvement in the pick wave makespan value under random slotting with 25ft^3 of the carton capacity and (1:10) of the ratio of carton capacity to the mean order volume means that we can save almost 18.57 hours (103532.0 - 36672.5 = 66859.5 seconds) by changing cartonization method from NFDV to NFDP-WZ. In these tables, one can find the relative percent improvement between NFDV and NFDP-WZ become larger, as the carton capacity and the ratio of carton capacity to the mean order volume increase. However, the difference of the number of cartons decreases and become similar each other between the heuristics as the carton capacity and the ratio of carton capacity to the mean order volume increase. The result indicates that NFDP-WZ is able to assign more line-items in the same zone to same carton as the mean order volume becomes larger than the carton capacity. Then the order can include more line-items in a carton and NFDP-WZ is also able to contain more line-items with the close proximity within zone, as the number of line-items per order becomes larger. Therefore, the relative percent improvement between NFDV and NFDP-WZ becomes larger. Furthermore, NFDV using the line-items in decreasing order of volume and NFDP-WZ using the line-items in decreasing order of proximity becomes a small difference in the number of cartons, as the number of line-items per order and/or the volume of order increase.

Table 4.5 Performance comparison between NFDV and NFDP-WZ

(a) Random slotting

CP(ft^3)		RT=(1:1)		RT=(1:3)		RT=(1:5)		RT=(1:10)	
		PI	DNC	PI	DNC	PI	DNC	PI	DNC
6.25	Mean	1.3	51	8.1	32	32.6	10	48.8	2
	SD	4.7	6	4.1	5	3.2	8	2.6	4
12.50	Mean	3.6	30	21.3	11	47.2	1	60.1	0
	SD	3.7	6	5.1	7	2.2	4	1.7	2
18.75	Mean	6.9	19	27.2	3	52.8	0	63.2	0
	SD	4.1	6	4.2	4	2.4	1	1.4	2
25.00	Mean	9.4	12	30.0	1	54.1	0	64.6	0
	SD	5.5	8	4.1	2	1.6	2	0.9	2

(b) COI slotting

CP(ft^3)		RT=(1:1)		RT=(1:3)		RT=(1:5)		RT=(1:10)	
		PI	DNC	PI	DNC	PI	DNC	PI	DNC
6.25	Mean	-1.3	51	5.5	31	26.3	5	40.2	1
	SD	2.6	4	3.1	5	1.6	3	2.0	3
12.50	Mean	3.3	30	14.5	8	39.6	1	53.6	1
	SD	2.8	4	2.5	3	1.6	2	0.9	3
18.75	Mean	2.4	16	18.7	3	45.1	0	57.3	0
	SD	2.4	3	2.5	2	1.1	1	1.0	2
25.00	Mean	1.6	8	20.4	1	48.8	1	61.1	1
	SD	2.4	3	2.6	2	1.0	1	0.8	2

The results of the performance comparison between NFDV and NFDP-WZ for COI slotting are presented in Table 4.5(b). The relative percent improvement and the difference between the number of cartons between NFDV and NFDP-WZ show similar results in Table 4.5(a). To compare Table 4.5(a) and 4.5(b), the relative percent improvement in COI slotting in Table 4.5(b) is worse than the relative percent improvement in random slotting in Table 4.5(a). As we earlier mentioned in the results in

Table 4.3, we can obtain high percent improvement between the NFDV and NFDP-WZ in poor slotting by showing a lot more reduction of the travel time. Meanwhile, in a sophisticated slotting, we are not able to obtain high percent improvement, because it already reduced the travel distance between the correlated line-items during the slotting process. Therefore, there is not much the relative percent improvement obtained by the cartonization (DNFP-WZ) with slotting information.

4.5 Conclusion

In this study, we considered the cartonization problem, which is the assignment problem for the line-items within orders to cartons. Since each carton directly ships to a customer after picking process, the line-items in the carton are assigned from a set of line-items in one order. We present a new mixed integer programming formulation to minimize pick wave makespan. Since the problem is NP-hard and the size of problem is very large, we propose a number of heuristic algorithms. Two types of cartonization heuristic methods without slotting information (FFDP-WZ, NFDP-WZ) and two types of cartonization heuristic methods with slotting information (FFDV, NFDV) are presented, respectively. The FFDP-WZ and NFDP-WZ provide a dominant pick wave makespan comparing to FFDV and NFDV, as the number of line-items in orders increase.

The cartonization problem becomes critical as mean order volume is larger than the carton capacity and the expected number of line-items per carton increases. In the number of line-items as 5 per order (LI), NFDV shows the lowest pick wave makespan and NFDP-WZ heuristic shows the lowest mean pick wave makespan as the number of line-items increases. Furthermore, NFDP-WZ heuristic requires more cartons from 2.4% (= 43/1697x100) to 2.5% (= 41/1693x100) than FFDV in the number of line-items per order (LI) as 25 (the largest number of line-items we tested). The relative percent improvement between NFDV and NFDP-WZ becomes larger as the carton capacity and/or the ratio of carton capacity to the mean order volume increase. This result indicates that NFDP-WZ shows better performance than NFDV as the containable line-items per carton become larger and the volume of order become larger than the carton capacity. The relative percent improvement between NFDV and NFDP-WZ is shown from -1.3% to 64.6%. The high percent improvement between the cartonization method without slotting information (NFDV) and the cartonization method with slotting information (NFDP-WZ) is shown in random slotting (RS) compared to COI slotting. RS potentially obtain more pick wave improvement by cartonization than COI, because RS is slotted without using any information of demand. Therefore, we can find that the slotting methods provide a significant impact for the performance of pick wave makspan.

Chapter 5
Iterative Slotting and Cartonization Method under Dynamic Warehouse Replenishment

Under the replenishment of the entire warehouse for a specific pick wave, slotting and cartonization should be decided at the same time. In this chapter, we propose a systematic slotting and cartonization method based on the slotting methods in chapter 3 and the cartonization methods in chapter 4 in zone-based carton picking system.

The rest of chapter 5 is organized as follows. In Section 5.1, we develop a nonlinear mixed integer programming (NL-MIP) slot assignment model for zone-based carton picking system. Then, three heuristics algorithms are proposed in Section 5.2. We report the results of computational experiments and analyze the performance of proposed heuristics in Section 5.3. Finally, we conclude the study with a summary and discuss some directions for future research in Section 5.4.

5.1 NL-MIP model for slotting and cartonization problem

In this section, we develop a nonlinear mixed-integer programming (NL-MIP) formulation for determining the assignment of SKUs into slots and the grouping of orders into cartons for a specific pick wave in a zone-based carton picking system. The general subscripts, parameters, variables are already explained in Section 3.2 and Section 4.2.

There are two primary decision variable sets in this formulation. First primary variable set is the slotting variable set, which decides the SKU to slot assignment. This variable set is shown in the decision variable set in chapter 3. The other one is the cartonization variable set, which decides the sepcific line-items grouped into the same carton. This variable set is shown in the decision variable set in chapter 4. The two primary decision variable sets are defined as follows:

x_{kmn} : indicator variable set, which is equal to 1 if SKU k is assigned to slot n in zone m; and 0 otherwise

q_{ijk} : number of SKU k in order *i* assigned to carton *j*.

The remaining variables depend on the value of x_{kmn} and q_{ijk}.

The completion time for a picker assigned to zone *m* is as follows ignoring starvation time as described in Section 1.1:

$$p_m = S_m + \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{k=1}^{K}\sum_{n=1}^{N} P_n x_{kmn} q_{ijk} + \sum_{j=1}^{J} d_{jm}, \qquad \text{for } \forall m \qquad (5.1)$$

The nonlinear 0-1 mixed integer model for the cartonization (NL_MIP) is formulated as follows:

$$\text{(NL\_MIP): min } t \qquad (5.2)$$

subject to:

$$\sum_{m=1}^{M}\sum_{n=1}^{N} x_{kmn} = 1, \qquad \text{for } \forall k \qquad (5.3)$$

$$\sum_{k=1}^{K} x_{kmn} \le 1, \qquad \text{for } \forall m, n \qquad (5.4)$$

$$\sum_{j=1}^{J} u_{ij} \geq 1, \qquad \text{for } \forall i \qquad (5.5)$$

$$\sum_{i=1}^{I} u_{ij} \leq 1, \qquad \text{for } \forall j \qquad (5.6)$$

$$\sum_{k=1}^{K} c_{ijk} \leq M u_{ij} \qquad \text{for } \forall i, j \qquad (5.7)$$

$$\sum_{k=1}^{K} c_{ijk} \geq u_{ij}, \qquad \text{for } \forall i, j \qquad (5.8)$$

$$\sum_{j=1}^{J} q_{ijk} = Q_{ik}, \qquad \text{for } \forall i, k \qquad (5.9)$$

$$\sum_{i=1}^{I} \sum_{k=1}^{K} V_k q_{ijk} \leq V, \qquad \text{for } \forall j \qquad (5.10)$$

$$q_{ijk} \leq M c_{ijk}, \qquad \text{for } \forall i, j, k \qquad (5.11)$$

$$(CS + 2W_n) \sum_{i=1}^{I} \sum_{k=1}^{K} c_{ijk} X_{kmn} \leq d_{jm}, \qquad \text{for} \forall j, m, n \qquad (5.12)$$

$$p_m \leq t, \qquad \text{for } \forall m \qquad (5.13)$$

$$p_{m+1} \leq p_m, \qquad \text{for } \forall m \setminus M \qquad (5.14)$$

$$x_{kmn,} u_{ij,} c_{ijk} \in \{0,1\}, \qquad \text{for } \forall i, j, k, m, n \qquad (5.15)$$

$$q_{ijk} \in \mathbf{N_0} \qquad \text{for } \forall i, j, k \qquad (5.16)$$

$$d_{jm}, p_m, t \geq 0, \qquad \text{for } \forall j, m \qquad (5.17)$$

In this formulation, we have four classes of the main constraint sets. The first class of constraint sets is the total picking processing time of picker constraints, second one is the slotting constraint set, third one is cartonization constraint set, and last one is carton capacity constraint set. The main constraint sets are partially introduced in the MIP models in chapters 3 and 4. Constraint set (5.3) ensures that each SKU is assigned to exactly one slot. Constraint set (5.4) ensures that each slot contains at most one SKU. Constraint set (5.5) ensures that an order must be assigned to at least one carton. Constraint set (5.6) ensures that all SKUs assigned to a carton belong to the same order. Constraint set (5.7) and (5.8) ensure that two variables, u_{ij} and c_{ijk} are coupled. The constraint sets provide all the SKUs in a carton must belong to the same order. Constraint set (5.9) ensures that total number of units of a given SKU for an order must be equal to the required number of SKUs for that order. Constraint set (5.10) ensures that the total volume of line-items assigned to a carton from a certain order must be less than or equal to the carton capacity. Constraint set (5.11) ensures the coupling constraint which couples c_{ijk} and q_{ijk}. Constraint set (5.12) ensures that the total picking process time for a picker assigned in zone *m* for carton *j* is greater than or equal to the carton visiting set up time plus two times of the travel time to the slot assigned a SKU *k* in the carton *j*. Constraint set (5.13) ensures that the pick time per picker in zone *m* is less than pick wave makespan *t*. Constraint set (5.14) helps us to ignore the symmetry of solutions and reduces feasible solution search space. This constraint set forces the total picking processing time for picker in zone *m* to be greater than the time for picker in zone *m*+1. It eliminates alternative optimal solutions when zone size increases. Constraint set (5.15) and (4.16) indicate that the decision variables are 0-1 integer and non-negative integer. Constraint set (5.17) ensures the remaining variables are non-negative. There is a non-linear constraint set by multiplying the decision variable sets x_{kmn} and q_{ijk} in constraint (5.1) and this formulation contains two NP-hard problems, (slotting and cartonization), Therefore, it is necessary to develop heuristic to solve the large problems within a limited time constraint.

5.2 Heuristic algorithms

Since the slotting and cartonization problem in this chapter is NP-hard, it is generally impossible to find guaranteed optimal solutions for the case of large problems. Before we propose the heuristic algorithms, the basic slotting heuristic and cartonization heuristic are already mentioned in previous chapters. For development of heuristic methods in this chapter, we used the most efficient slotting improvement heuristic as the simulated annealing using the correlated interchange (SA-C) in chapter 3 and we used two cartonization heuristics in chapter 4. We used the next fit decreasing by volume (NFDV) as the cartonization heuristic without slotting information and we also used the next fit decreasing by proximity without zone separation (NFDP-WZ) as the cartonization with slotting information. Based on the slotting and cartonization heuristics, we develop three types of heuristic procedures for slotting and cartonization problem in this chapter.

In first heuristic, we first randomly assign SKUs into slots and then we next proposed NFDP-WZ heuristic based on the slotting information obtained by the random slotting. We called it as *slotting first and cartonization next* heuristic (*SFCN*). In second heuristic, we first assign line-items within an order to cartons using NFDV cartonization heuristic and then we next construct correlated list and perform COI slotting based on cartonization information obtained by NFDV heuristic. In second heuristic, we call it as *cartonization first and slotting next* heuristic (*CFSN*).

In the final heuristic, we first performed the random slotting and then we next proposed NFDP-WZ heuristic based on the slotting information obtained by the random slotting. Once the initial slotting is constructed, we iteratively reassign line-items into cartons using NFDP-WZ based on the slotting information in the previous stage and reassign SKUs into slots using SA-C slotting heuristic based on the cartonization information in the previous stage. We call it as *iterative approach on slotting and cartonization* heuristic (*ISC*). In *ISC*, the pick wave makespan decreases and converges to a stable pick wave makespan as the number of stages (a set of slotting heuristic and cartonization heuristic) increases, because the each heuristic improves the solution in slotting and cartonization information in the previous stage. The general heuristic procedure in three heuristics is shown in the following section.

5.2.1 *SFCN* heuristic

If there is no cartonization information, the random slotting policy is popular and general. Based on the random slotting information, we can propose a sophisticated cartonization method. In this heuristic, we randomly assign SKUs into slots and then we propose NFDP-WZ heuristic based on the random slotting information. The detailed NFDP-WZ heuristic procedure is described in Section 4.3.2 in Chapter 4.

Procedure: SFCN

Step 1: Use a random slotting (RS) as an initial slotting.

Step 2: Use a NFDP-WZ cartonization heuristic, based on the slotting information of SKUs by *RS* slotting.

Step 3 Calculate pick wave makespan in an arbitrary order of cartons.

5.2.2 *CFSN* heuristic

If there is no slotting information of SKUs, the NFDV cartonization method is one of efficient cartonization methods. It reduces the number of picks per cartons by assigning as many items as possible into a carton. In this case, there is a potential reduction of picking travel time within a zone and carton set up time between the line-items per carton. However the NFDV cartonization method may contain additional zones or additional farthest slots within a zone by including the items into a carton until the capacity of the carton is reached and it potentially results in a higher pick wave makespan by increasing total picking time of one or more pickers. Therefore, we need more sophisticated slotting method based on cartonization information of NFDV. In this heuristic, we first assign line-items within an order into carton using NFDV cartonization heuristic, and then we perform COI slotting based on the cartonization information. The detailed heuristic algorithms are described in Section 3.3 in Chapter 3 and Section 4.3.1 in Chapter 4.

Procedure: CFSN

Step 1: Use a NFDV as an initial cartonization.

Step 2: Generate a correlated list based on the NFDV cartonization and perform COI slotting based on the cartonization information by NFDV.

Step 3 Calculate pick wave makespan in an arbitrary order of cartons.

5.2.3 *ISC* heuristic

Neither *SFCN* nor *CFSN* heuristics guarantee a good solution because both heuristic approaches are basically assumed to be already determined in one problem without having any information and solve the other problem efficiently using the information on the first problem. If we decompose the slotting and cartonization problem into two independent problems and we iteratively solve one problem under the order problem being fixed, we can potentially identify more good solutions (Polito et al. 1980). This is the *ISC* heuristic.

In this heuristic, we first perform a random slotting as an initial slotting and then we propose NFDP-WZ cartonization heuristic based on the slotting information by the random slotting. Once the initial slotting and cartonization is constructed, we iteratively reassign SKUs into slots using SA-C slotting heuristic based on the cartonization information in the previous stage or line-items into cartons using NFDP-WZ based on the slotting information in the previous stage until the pick wave is converged and stable. The detailed heuristic algorithms are described in Section 3.3 and in Section 4.3.2.

Procedure: ISC

Step 1: Use a random slotting (RS) as an initial slotting.

Step 2: Use a NFDP-WZ cartonization heuristic, based on the slotting information of SKUs by RS slotting.

Step 3 Generate a correlated list based on the NFDP-WZ cartonization and perform SA-C slotting based on the cartonization information by NFDP-WZ.

Step 4 Calculate pick wave makespan in an arbitrary order of cartons. Go to Step 2 until termination time, N.

5.3 Computational results

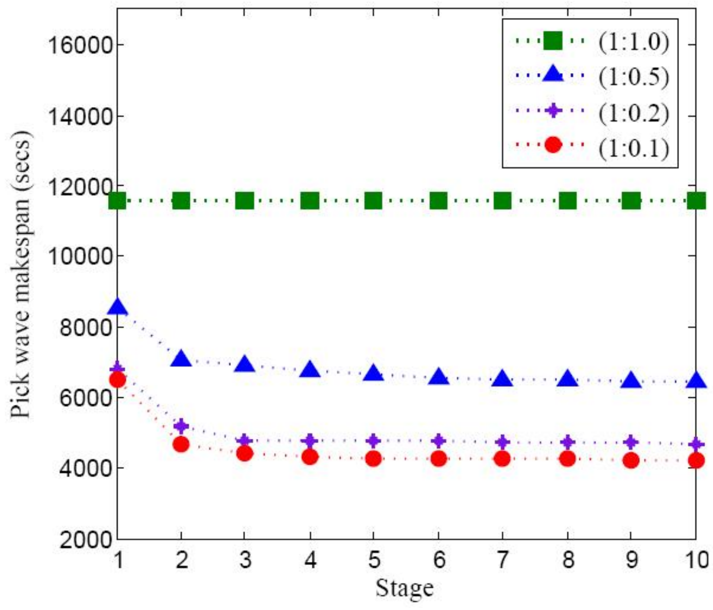
To evaluate the performance of *ISC* heuristic, we compare the results of the *SFCN* heuristic and C*FSN* heuristic. The order picking system parameters are described in table 3.1 in chapter 3. We fixed the carton capacity as 6.25 ft^3. The number of orders is fixed as 100. Since the performance of problem depends on the number of containable line-items per carton (LI), the ratio of the carton capacity to the mean order volume (RT), and the ratio of picking time to carton set up time, we control these three parameters. In this section, we first present the solution convergence of the *ISC* heuristic as the iteration of slotting and cartonization increases and we next examine the performance of the heuristics by changing several control parameters. To show consistent performance for each control parameters, we assume that the quantity of each line item in an order is 1. In each parameter set, we generated 10 random problems. The heuristic approaches have been coded in C++ and run on Pentium IV 2.0 GHz CPU with 2.0 GB memory.

5.3.1 Pick wave makespan convergence in *ISC* heuristic

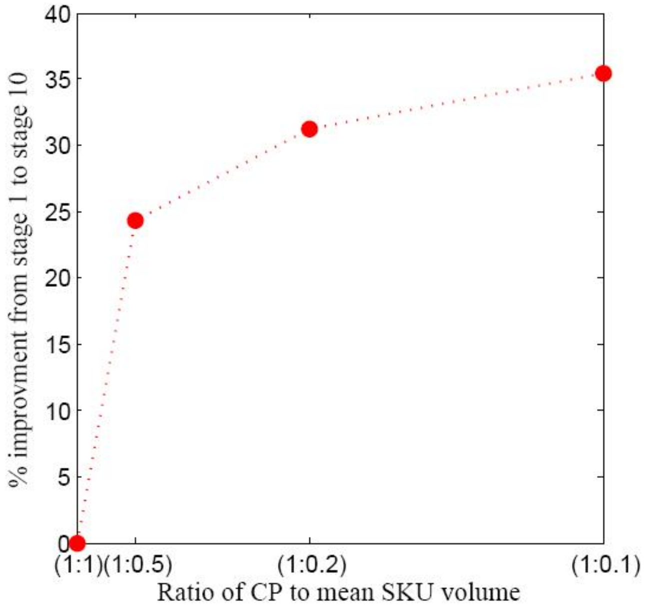
ISC heuristic is initiated by a random slotting as an initial slotting and then we propose NFDP-WZ cartonization heuristic based on the slotting information by the random slotting. Once the initial slotting and cartonization is constructed, we iteratively reassign SKUs into slots using SA-C slotting heuristic based on the cartonization information in the previous stage or line-items into cartons using NFDP-WZ based on the slotting information in the previous stage until the pick wave is converged and stable.

Figure 5.1(a) shows pick wave makespan convergence for the different ratio of carton capacity (CP) to the mean SKUs volume (i.e., (1:1), (1:0.5), (1:0.2), and (1:0.1)) and percent improvement from stage 1 to stage 10 is plotted at each ratios in Figure 5.1 (b). Since the carton capacity is fixed as 6.25ft^3, the mean SKUs volume are 6.25, 3.125, 1.25, and 0.625ft^3 for (1:1), (1:0.5), (1:0.2), and (1:0.1), respectively. In Table 5.1(a), the pick wave makespan decreases and converges to a low point in each ratio of carton capacity to the mean SKU volume except (1:1), as the stage increases. In (1:1), there is no improvement as stage increases because only one line-item can be assigned in each carton because of carton capacity. Therefore, pick wave makespan cannot be reduced by both slotting method and cartonization method. If carton capacity is greater than the mean SKU volume, the number of cartons is reduced because more items can be assigned into a carton. Thus, an initial pick wave makespan decreases as the ratio is small. From stage 1 to stage 2, the pick wave makespan quickly decreases, because SA-C slotting in stage 2 is more intelligent slotting method than random slotting in stage 1. From second stage, the pick wave makespan of slotting and cartonization is shown in decreasing trend as the stage increases, because the both heuristics improves solution based on the previous slotting or cartonization domain. However, the decrement of the pick wave makespan by SA-C slotting becomes quickly small as the stage increases.

In this study, SA-C improvement is highly depends on the number of correlated SKU pairs. Due to intelligent cartonization heuristic, the number of correlated SKU pairs becomes large as the stage increases. However, the increased number of the correlated interchange in SA-C is not able to obtain much improvement in the proportions to the number correlated SKU pairs, because most of the correlated SKUs pairs in SA-C in the current stage are already assigned in next to each other from the SA-C in the previous stages. Therefore, the pick wave makespans for each ratio are shown in the convergence to a lowest pick wave makespan until the stage 10. In table 5.1(b), the effect of the skewness for the percent improvement from stage 1 to stage 10 is diminished, as the ratio become small (more items per cartons). *ISC* heuristic shows almost 35% improvement compared *SFCN* heuristic in (1:0.1).



(a) Pick wave convergence in ISC heuristic for different ratios of carton capacity (CP) to mean SKU volume



(b) Percent improvement from stage 1 to stage 10 for different ratios from carton capacity (CP) to mean SKU volume

Figure 5.1 Pick wave makespan convergences in ISC heuristic ((A:B) = the ratio of carton capacity to mean SKU volume)

5.3.2 Performance comparison of heuristics for containable line-items per carton

Several researchers have studied the relation between pick-density and storage assignment rule (Hall 1993, Carton, 1998, Hwang, 2004) in the multiple picks per pick tour case. From their results, they showed the reduction of travel distance/time as the number of picks per picking tour increases and also showed more the reduction of travel distance/time when the SKUs are slotted in a sophisticated slotting method. The limitation of their studies is that they only performed COI slotting based on an analytical model using the different shaped ABC curve. Since we have shown that the SA-C slotting using correlated interchange is outperformed than the COI slotting in chapter 3, we test the performance for the relation between pick-density and three heuristics using different slotting methods.

Since the number of items per carton is a decision variable in this study, we indirectly control the number of containable items per carton by adjusting the mean volume of each SKU under the fixed carton capacity. This works because the proposed cartonization heuristic is based on bin packing heuristic and it should assign as many line-items as possible to a carton. In this section, we first randomly generate pick-waves with 5, 10, 15, 20, 25, 30, 40, 50, 70, and 100 line-items per order. Then we have to set the mean volume of SKUs for each line-item. Since the carton capacity is fixed 6.25ft^3 and the number of line-items per order is controlled, the mean SKU volume can be set as 1.250ft^3(=6.25/5), 0.625ft^3(=6.25/10), 0.416ft^3(=6.25/15), 0.312ft^3(=6.25/20), 0.250ft^3(=6.25/25), 0.208ft^3(=6.25/30), 0.156ft^3(=6.25/40), 0.125ft^3(=6.25/50), 0.089ft^3(=6.25/70), and 0.0625ft^3(=6.25/100) so that all the line-items for an order will fit into a single carton. Therefore, we can isolate the effect of the slotting heuristic by minimizing the effect of the cartonization heuristic.

Figure 5.2 shows the performance of heuristics for the number of containable items per carton. In general, three heuristics have pick wave makespan reduction, as the number of containable items per carton increases. Initially, the slope quickly decreases because we obtain a potential travel time reduction of pickers by assigning items in near slots to the same carton. If the number of containable items per carton is large, the slope of pick wave makespan changes to be constant. In this case, all the cartons visit all zones so that the carton set up time reduction could not critical effect on the pick wave makespan of pickers. The pick wave makespan only depends on the picking time and walking time of picker within each zone. The number of items per zone increases, as the number of containable items per carton increases. Therefore, the pick wave makespan almost linearly increases in large number of containable items per carton, as number of items per zone increases. In this figure, *ISC* heuristic shows the lowest pick wave makespan in all the number of containable line-items per carton. Since we minimize the effect on cartonization by assigning all the line-items in an order can assign only one carton, the pick wave makespan of each heuristic is reduced in the order of random slotting (RS), COI slotting, and SA-C slotting at any number of containable items per carton.

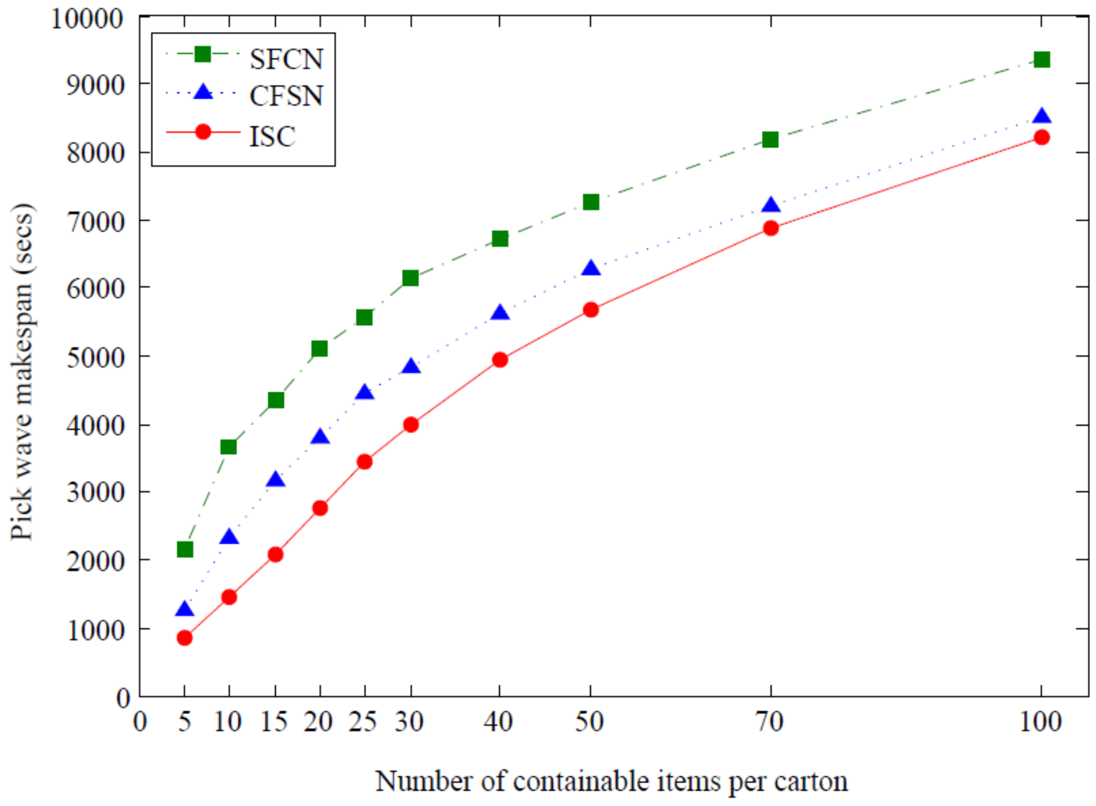


Figure 5.2 Performance comparisons of heuristics for the number of containable items per carton

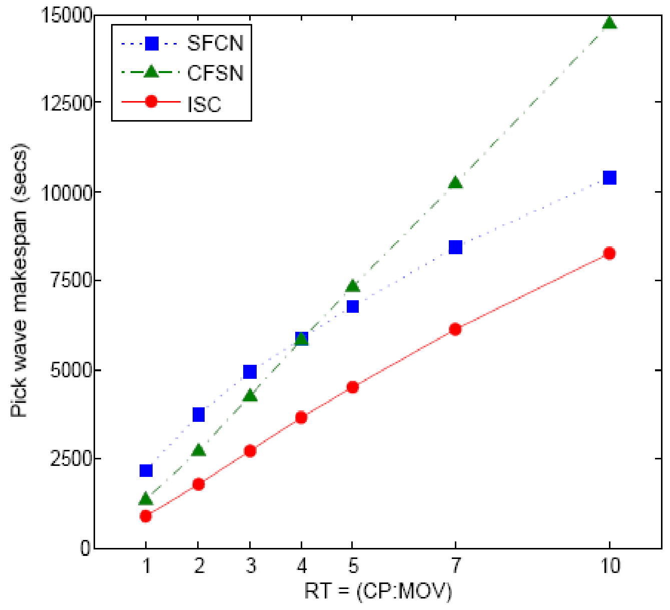
The reduction of pick wave makespan in the *ISC* heuristic is relatively diminished as the number of containable items increases compared to *SFCN* heuristic (RS+NFDP-WZ) and *CFSN* heuristic (NFDV+COI). *ISC* heuristic uses a number of SKU pairs in the correlated list for SA-C slotting. The number of SKU pairs in the correlated list becomes extremely large as the number of line-items increases. Then, the exploring space in SA-C slotting also becomes dramatically large. Therefore, the large exploring space results in poor performance, even if the potential improvement is still existed.

5.3.3 Performance comparison of heuristics for the ratio of the carton capacity to the mean order volume

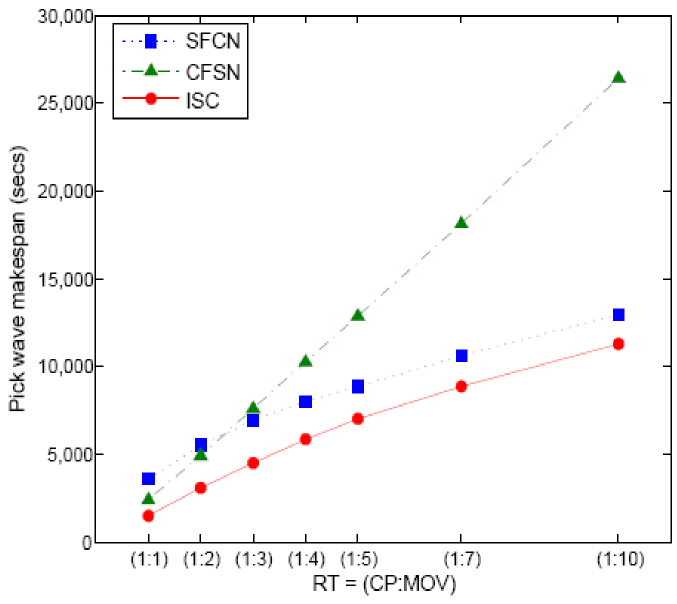
The cartonization has a critical effect on the performance of the solution when the order volume is greater than carton capacity. For generating the mean order volume, we first set the mean unit volume of SKUs as a constant value and then we control the number of line-items per order. The ratio of the carton capacity to the mean order volume (RT) is defined as $(i:j)$, such that an order can be contained in $\lceil j/i \rceil$ cartons. For example, if we want to set RTs as (1:1), (1:2), (1:3), (1:4), (1:5), (1:7), and (1:10), we first set the mean SKUs as a constant value, 1.25ft^3 and we already fixed the carton capacity as 6.25ft^3. Next, we control the number of line-items per order as 5, 10, 15, 20, 25, 35, and 50, respectively. Then, the right-hand values of RT are determined as 1(=5x1.25/6.25), 2(=10x1.25/6.25), 3(=15x1.25/6.25), 4(=20x1.25/6.25), 5(=25x1.25/6.25), 7(=35x1.25/6.25), and 10(=50x1.25/6.25), respectively. If we set the mean SKUs as a constant value, 0.625ft^3, we can obtain same RTs by increasing twice number of line-items per order, 10, 20, 30, 40, 50, and 70, respectively.

[1(=10x0.625/6.25), 2(=20x0.625/6.25), 3(=30x0.625/6.25), 4(=40x0.625/6.25), 5(=50x0.625/6.25), 7(=70x0.625/6.25), and 10(=100x0.625/6.25)]. Then, we can generated same RTs as (1:1), (1:2), (1:3), (1:4), (1:5), (1:7), and (1:10) for both mean SKU volumes of 1.25ft^3 and 0.625ft^3.

In Figure 5.3(a) and 5.3(b), the mean pick wave makespans for three heuristics are plotted for different RTs in the mean SKUs volume of 0.625 ft^3 and 0.125ft^3. In this figure, *ISC* shows the lowest pick wave makespan for all RTs. The difference of the pick wave makespan between *CFSN* (RS+NFDP-WZ) and *ISC* increases as RT increases. Since *CFSN* heuristic performs cartonization without slotting information and *ISC* heuristic performs cartonization using slotting information, the difference of pick wave makespan between *CFSN* and *ISC* becomes large as RT becomes high. The percent improvement between *CFSN* and *ISC* varies from 34.9% to 57.6% and the percent improvement between *SFCN* and *ISC* varies from 13.0% to 60.0%. In Figure 5.3(a), the mean SKU volume is 1.250 ft^3. Since carton capacity is fixed as 6.25ft^3, there are approximately 5 items can be included in a carton. In Figure 5.3(b), there are approximately 10 items can be included in a carton because the mean SKU volume is 0.625ft^3. If the number of line-items per carton increases, carton picking tour time increases. Therefore pick wave makespan in Figure 5.3(b) is almost twice time higher than Figure 5.3(a). To compare the performance of pick wave makespan for *SFCN* and *CFSN* graphs in Figure 5.3(a) and 5.3(b), *CFSN* shows less pick wave makespan than *SFCN* in low RT. Meanwhile, *SFCN* shows less pick wave makespan than *CFSN* in high RT.



(a) Mean SKU volume: 1.250 ft^3



(b) Mean SKU volume: 0.625 ft^3

Figure 5.3 Performance comparison of heuristics for the ratio of the carton capacity to the mean order volume (CP: carton capacity, MOV: mean order volume)

This result indicates the slotting heuristic has a critical effect on the performance of the pick wave makespan in the case that there is small difference between mean order volume and carton capacity and the cartonization heuristic becomes critical on the performance of the pick wave makespan in the case that the order volume is larger than carton capacity. As the number of containable items per carton increases by decreasing the mean SKU volume, *SFCN* shows dominant pick wave makespan than *CFSN* at almost the entire ratios, because the pick-density is increased and NFDP-WZ in *SFCN* can assign more items slotted in near slots in the same zone into same carton than NFDV in *CFSN*.

5.3.4 Performance comparison of heuristics for the ratio of picking time to carton set up time

The performance of pick wave makespan depends on carton set up time and picking time and the number of picks per cartion (pick density). To include the effect on both slotting heuristic and cartonization heuristic, we first adjust the mean volume of SKUs for each line-items per order (i.e.,1.250ft^3 (=6.25/5), 0.625ft^3 (=6.25/10), 0.416ft^3 (=6.25/15), 0.312ft^3 (=6.25/20), 0.250ft^3 (=6.25/25), 0.208ft^3 (=6.25/30), 0.156ft^3 (=6.25/40), 0.125ft^3 (=6.25/50), 0.089ft^3 (6.25/70), and 0.0625ft^3(6.25/100)) and then we can assign 5, 10, 15, 20, 25, 30, 40, 50, 70, and 100 items into a carton, respectively. Next, we generate pick waves with the number of line-items per order as 3 times larger than carton capacity. (i.e., 15(=3x5), 30(=3x10), 45(=3x15), 60(=3x20), 75(=3x25), 90(=3x30), 120(=3x40), 150(=3x50), 210(=3x70), and 300(=3x100)). Then we can include the effect on both slotting and cartonization. The ratio of the mean picking time to carton set up time is defined into $(i:j)$, where $(i:j)$ means the carton set up time is j/i time longer than the mean picking time.

Figure 5.4 shows the performance of pick wave makespan of *ISC* heuristic for different ratio of picking time and carton set up time. The pick wave makespan increases as the ratio increases. In all the ratio of the mean picking time to carton set up time, there is a significant reduction of pick wave makespan between 30 and 90 of the number of line-items per order (the number of containable item per carton between 15 and 30) at each ratio graph in 10 zone carton picking system.

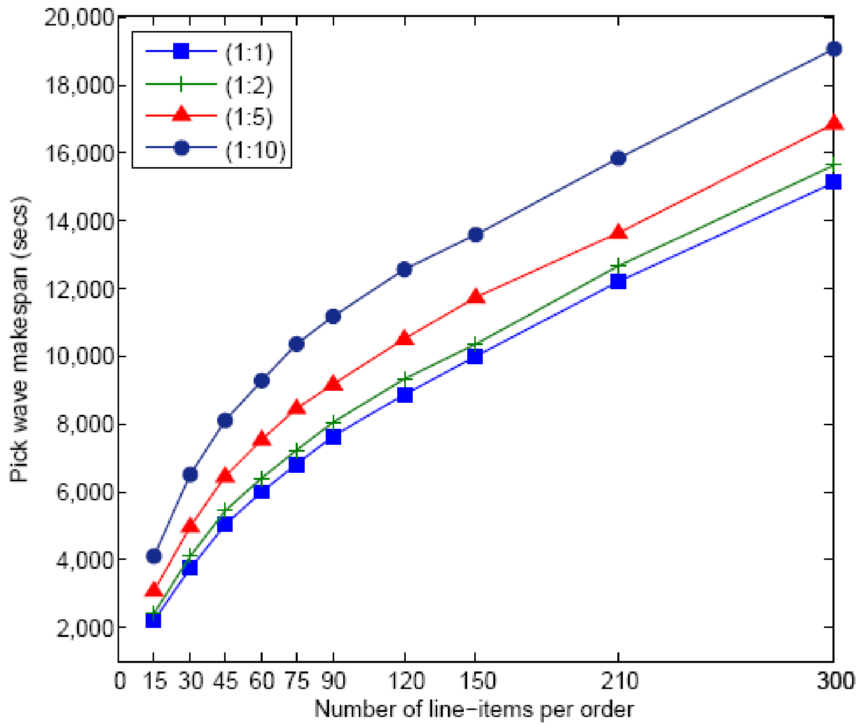


Figure 5.4 Performance comparisons of heuristics for different ratio of picking time to carton set up time, (A:B) = the ratio of pick time (A) to carton set up time (B)

5.4 Conclusions

In this study, the distribution center has different sets of SKUs being picked on short-term period or different days. Thus, the entire picking area is periodically reslotted, and in the target environment the periods are typically quite short (e.g., one day). The decision for an efficient slotting depends on the decision for an efficient cartonization of a pick wave, and vice versa. Therefore, the decisions for the slotting and the cartonization should be solved simultaneously. Since solving the slotting problem (or the cartonization) under the predetermined cartonization (or the slotting) being given does not guarantee a good solution, it is necessary to develop the two problems simultaneously to improve the solution in a dynamic whole warehouse replenishment environment. In chapter 5, we proposed an iterative slotting and cartonization heuristic using the slotting heuristic procedure in chapter 3 and cartonization heuristic procedure in chapter 4.

We proposed three types of heuristics in this chapter. In the first heuristic, we propose the *slotting first and cartonization next* heuristic (*SFCN*). In this heuristic, we randomly assign SKUs into slots and then we proposed NFDP-WZ heuristic based on the slotting information. In second heuristic, we proposed the *cartonization first and slotting next* heuristic (*CFSN*). In this heuristic, we assign line-items within an order into carton using NFDV cartonization heuristic, and then we perform COI initial slotting. In final heuristic, we proposed the *iterative approach on slotting and cartonization* heuristic (*ISC*). In this heuristic, we first perform a random slotting as an initial slotting and then we next proposed NFDP-WZ heuristic based on the slotting information obtained by the random slotting. Once the initial slotting and cartonization is constructed, we iteratively reassign SKUs into slots using SA-C slotting heuristic based on the cartonization information in the previous stage or line-items into cartons using NFDP-WZ based on the slotting information in the previous stage until the pick wave is converged and stable.

In this chapter, we present several interesting testing results. First, we showed *ISC* solution decreases the pick wave makespan quickly converged, as the number of stages increases. The percent improvement from first stage to tenth stage increases as the ratio of carton capacity to the mean SKU volume increases. However the increment of the percent improvement becomes small as the ratio becomes large. Second, *ISC* heuristic shows the lowest pick wave makespan at the various the numbers of containable line-items per carton. Since we minimize the effect on cartonization by assigning all the line-items in an order to one carton, the pick wave makespan of each heuristic is reduced in order of random slotting (*SFCN*), COI slotting (*CFSN*), and *SA-C* slotting (*ISC*) at any containable line-items per carton. Third, *ISC* heuristics shows better performance than *SFCN* and *CFSN* heuristics at the various ratio of the carton capacity to the mean order volume. In this test, the slotting heuristic is a critical effect on the performance of the pick wave makespan in the case that there is small difference between order volume and carton capacity and the cartonization heuristic becomes a critical effect on the performance of the pick wave makespan in the case that the order volume is larger than carton capacity. Last, the pick wave makespan linearly increases as the carton set up time increased at any given number of line-items per order. The largest pick wave reduction is shown in the number of line-items per order between 30 and 90 (the number of containable item per carton between 15 and 30) at each ratio graph. Overall, *ISC* showed outperformed performance than *SFCN* and *CFSN* heuristics in various control parameters.

Chapter 6
Conclusions and Future Research

Warehouses are essential components to reduce logistics cost in a supply chain. In this dissertation, two warehouse operations (slotting and cartonization) are considered in a zone-based carton picking system. The slotting operation is determining an assignment of SKUs to picking slots to support the order picking system. This operation is essentially the same as the storing operation. The cartonization operation is determining an assignment of line-items within an order to cartons with a limited capacity. In the target warehouse, different sets of SKUs are picked on different days of the week and the picking area is short-term periodically re-slotted for each pick wave. Under the dynamic whole warehouse replenishment environment, without the both decisions for slotting and cartonization, the order picking cost is not able to construct in the zone-based picking system. The problems studied in this dissertation, therefore, are related with both the slotting and cartonization operations affecting to order picking cost in the zone-based carton picking system. Before we proposed a model for both problems, we regard the problems as independent one and solved the problem separately. Two MIP formulations for slotting and cartonization are proposed. Since both problems are independently NP-hard, we proposed an efficient heuristics, respectively. In slotting problem, we developed a simulated annealing using correlated list (SA-C). It provided a good performance in large problems under limited planning time. In the cartonization problem, we proposed a bin-packing based heuristic considering slotting information. It showed the good performance as the number of line-items per order and the ratio of order volume to the carton capacity increase. Once we developed the independent models for the slotting and the cartonization, we finally proposed a systematic iterative heuristic model based on the independent models to control the both two NP-hard problems (i.e., slotting and cartonization) for a dynamic pick-wave.

Several directions for future research are apparent from this dissertation. The current study in this dissertation is confined as zone-based carton picking system.

First, we need to extend the study to more generalized order picking system (i.e. manual pick and walk picking system) for controlling dynamic replenishment environment. In that case, we need to consider travel routing to estimate order picking cost. There are several studies that focus on the evaluation of the routing and slotting policies in manual pick and walk order picking systems. However, the slotting policies considered are generally limited to random or COI-based slotting. In this dissertation, we presented that the correlated slotting (CS) is better performance than COI slotting in the zone-based carton picking system. However, the results on this study are limited in the zone-based carton picking system. Therefore, we need to generalize the order picking system for the performance of CS slotting compared to COI slotting. There are several things to be considered for the extension of the generalized order picking system. First, if the study extends to the generalized order picking system (i.e. manual pick and walk picking system), routing problem and congestion problem should be considered. Second, we should develop a pick wave generation considering the turn-over rate of SKUs in the picking system. Once we generate the pick wave, we can compare the performance of CS with COI for a specific pick wave with the turnover items. In the last, if we can develop a closed-form expression for representing correlated slotting, we can develop an analytical model and probabilistic analysis for the CS slotting in multiple picking. The analysis on solutions of the model would be great impact on many order-picking systems or other applications.

Second, in the slotting area in this dissertation, we did not consider cases where a SKU must be assigned to multiple slots (i.e., where total number of units demanded in the pick-wave exceeds the capacity of a single slot in the pick area). The difficulty here is that the criteria used to select a particular slot for a particular carton have not been incorporated into the mathematical formulation and we have an additional decision for which item ordered is selected to one of slots.

Third, in the cartonization problem, there are several extensions for the future research. As one can see in the problem description and results, this problem is significantly more difficult when line-items per carton and the ratio of carton capacity to the mean order volume become larger. In this situation, we need meta-heuristics to find a near optimal solution within a reasonable computing time. The heuristics in this study shows a good performance consuming the reasonable number of cartons comparing the number of cartons using classical bin-packing problem. Therefore, we need to consider the multi-objective functions (i.e, the pick-wave makespan and the number of cartons consumed to satisfy the makespan, for this problem). Finally, we found the performance of cartonization is highly interrelated with the slotting methods. Therefore, both problems should be solved simultaneously to achieve the improvement of further solution performance.

Finally, in dynamic slotting in this dissertation, we currently neglect relocation cost, because the target order picking system replenishes the entire picking area at every period. If the order picking system is in the dynamic partial slotting environment, the order picking cost model should be changed. There is no literature considering both replenishment cost and relocation cost under a specific pick-wave.

References

Bartholdi, J.J., Hackman, S.T. (2008) Allocating space in a forward picking area of a distribution center for small parts. *IIE Transaction*, **40**, 1046-1053.

Bartholdi, J.J., Hackman, S.T. (2008) *Warehouse & distribution science*. Available on line at: http://www2.isye.gatech.edu/~jjb/wh/book/editions/wh-sci-0.89.pdf.

Bassan, Y., Roll, Y., and Rosenblatt, M.J. (1980) Internal layout design of a warehouse. *AIIE Transactions*, **12**(4), 317−322.

Bozer, Y.A. and Carlo, H.J. (2008) Optimal inbound and outbound door assignments in less-than-truckload crossdocks. *IIE Transactions*, **40**, 1007-1018.

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

Brynzer, H. and Johansson, M.I. (1996) Storage location assignment: using the product structure to reduce order picking times. *International Journal of Production Economics*, **46**, 595−603.

Bukard, R.E. (2002) Selected topics on assignment problems. *Discrete Applied Mathematics*, **123**(1-3), 257-302.

Bukard, R.E. and Rendl, F. (1984) A thermodynamically motivated simulation procedure for combinatorial optimization problems, *European Journal of Operational Research*, **17**, 169-174.

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

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

Chew, E.P., Tang, L.C. (1999) Travel time analysis for general item location assignment in a rectangular warehouse. *European Journal of Operational Research*, **112**, 582−597.

Chisman, J.A. (1975) The clustered travelling salesman problem, *Computers & Operations Research* **2**, 115-119.

Chisman, J.A. (1977) Optimizing the shipping function, *Journal of Industrial Engineering*, **9**, 38-41.

Choe, K. and Sharp, G. P. (1991) Small parts order picking: design and operation, available on-line at: http://www2.isye.gatech.edu/logisticstutorial/order/article.htm.

Christofides, N. and Colloff, I. (1973) The rearrangement of items in a warehouse. *Operations Research,* **21**, 577–589.

Cormier, G. (1987) On the scheduling of order-picking operations in single-aisle automated storage and retrieval systems. In: Kusiak, A. (Ed.), Modern Production Management Systems. Elsevier Science Publishers, pp. 75–87.

Coyle, J.J., Bardi, E.J. and Langley, C.J. (1996) The management of business logistics, Thomson West: Mason, OH.

Council of Supply Chain Management Professionals (2008) The 19th Annual State of Logistic Report.

De Koster, R. and Yu, M. (2008) Minimizing makespan and throughput times at Alsmeer flower auction. *Journal of Operational Research Society*, **59**, 1182-1190.

De Koster, R. (1994) Performance approximation of pick-to-belt orderpicking systems. *European Journal of Operational Research*, **72**, 558–573.

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.

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

De Koster, R., Roodbergen, K.J., and Van Voorden, R. (1999) Reduction of walking time in the distribution center of De Bijenkorf. In: Speranza, M.G., Sta¨hly, P. (Eds.), New Trends in Distribution Logistics. Springer, Berlin, pp. 215–234.

Drury, J. (1988) Towards more efficient order picking. IMM monograph no. 1, The Institute of Materials Managements: Cranfield, UK.

Elsayed, E.A. (1981) Algorithms for optimal material handling in automatic warehousing systems. *International Journal of Production Research*, **19**(5), 525−535.

Elsayed, E.A., Lee, M.-K. (1996) Order processing in automated storage/retrieval systems with due dates. *IIE Transactions*, **28**(7), 567−577.

Elsayed, E.A., Lee, M.-K., Kim, S., 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.

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

Elsayed, E.A., Unal, O.I. (1989) Order batching algorithms and travel-time estimation for automated storage/retrieval systems. *International Journal of Production Research,* **27** (7), 1097−1114.

Francis, R.L. (1967) On some problems of rectangular warehouse design and layout. *Journal of Industrial Engineering*, **18**(10), 595−604.

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

Frazelle, E.A. (1990) Stock location assignment and order picking productivity, MHRC-TD-89-11, Material Handling Center, Atlanta, Georgia.

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

Frazelle, E.H., Hackman, S.T., Passy, U., Platzman, L.K. (1994) The forward-reserve problem. In: Ciriani, T.A., Leachman, R.C. (Eds.), Optimization in Industry, vol. 2. John Wiley & Sons Ltd., New York.

Gademann, A.J.R.N., Van den Berg, J.P., and Van der Hoff, H.H. (2001) An order batching algorithm for wave picking in a parallel-aisle warehouse. *IIE Transactions*, **33**, 385−398.

Gademann, A.J.R.N.. and Van de Velde, S. (2005) Batching to minimize total travel time in a parallel-aisle warehouse. *IIE Transactions,* **37**(1), 63−75.

Garey, M.R., Johnson, D.S. (1979) Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman, San Francisco.

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

Goetschalckx, M. and Ratliff, H. D. (1990) Shared storage policies based on the duration of stay. Management Science, 36(9), 1120-1132.

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

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

Graves, S.C., Hausman, W.H., and Schwarz, L.B. (1977) Storage/retrieval interleaving in automatic warehousing systems. Management Science, 23, 935−945.

Gu, J. 2005. The forwared reserve warehouse sizing dimensioning problem, PhD thesis, Geogia Institue Technology, the united state.

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

Gue, K. R. and Meller, R.D. (2009) Aisle Configurations for Unit-Load Warehouses. IIE Transactions, 43(3), 171-182.

Gue, K.R., 2006. Very high density storage systems, IIE Transactions, 38, 93−104.

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

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.

Hall, R.W. (1993) Distance approximation for routing manual pickers in a warehouse. IIE Transactions, 25, 77−87.

Harmatuck, D.J. (1976) A comparison of two approaches to stock location, Logistics and Transportation reviews, 12(4), 282-284.

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. *Transport and Distribution Management*, **3**, 27−31.

Heskett, J.L. (1964) Putting the cube-per-order index to work in warehouse layout. *Transport and Distribution Management*, **4**, 23−30.

Herague, S.S. and Alfa, A.S. (1990) A hybrid simulated annealing based algorithm for the layout problem, *European Journal of Operational Research*, **53**, 1-13.

Ho, Y.-C. and Chen, M.C. (2006) A comparison of two zone-visitation sequencing strategies in a distribution centre. *Computers & Industrial Engineering*, **50**, 426-439.

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

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

Hsu, C.M., Chen, K.Y., and Chen, M.C. (2005) Batching orders in warehouses by minimizing travel distance with genetic algorithms. *Computers in Industry*, **56**(2), 169−178.

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

Hwang, H., Baek, W., Lee, M.-K. (1988) Clustering algorithms for order picking in an automated storage and retrieval system. *International Journal of Production Research*, **26**(2), 189−201.

Hwang, H., Lee, M.-K. (1988) Order batching algorithms for a man-on-board automated storage and retrieval system. *Engineering Costs and Production Economics*, **13**, 285−294.

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

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

Hwang, H., Yong, H.O., Cha, C.N. (2003) A stock location rule for a low level picker-to-part system. *Engineering Optimization*, **35**(3), 285–295.

Jaikumar, R. and Solomon, M.M. (1990) Dynamic operational policies in an automated warehouse. *IIE Transactions*, **22**(4), 370–376.

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

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

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

Johnson, D.S. (1973) Near-optimal bin packing algorithms, Ph.D. thesis, Massachusetts Institue of Technology, Department of Mathematics, Cambridge.

Johnson, D.S. (1974) Fast algorithms for bin packing. *Journal of Computer and System Science*, **8**, 272-314.

Johnson, D.S., Demers, A., Ullman, J.D., Garey, M.R., Graham, R.L. (1974) Worst-case performance bounds for simple one-dimensional packing algorithms. *SIAM Journal on Computing*, **3**, 299–325.

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.

Kirkpatrick, S., Gellatt, C.D. Jr., and Vecchl, M.P. (1983) Optimization by simulated annealing, *Science*, **220**, 671-680.

Kim, B.S. and Smith, J.S. (2008) Dynamic slotting for zone-based distribution center picking operation. *10th International Material Handling Research Colloquium*, Dortmund, Germany, 577-599.

Koopmans, T.C. and Beekman, M. (1956) Assignment problems and the location of economic activities. *Econometrica*, **25**(1), 53-76.

Landers, T.L. and Beaver, M.K., Sadiq, M., and Stuart, D.E. (1994) Software for dynamic re-configurable order picking systems. *Computers & Industrial Engineering*, **1-4**, 245-248.

Larson, T.N. and March, H., Kusiak, A. (1997) A heuristic approach to warehouse layout with class based storage. *IIE Transactions*, **29**, 337−348.

Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A. H. G., and Shmoys, D. B. (1995) The Traveling Salesman Problem. Wiley, Chichester.

Le-Duc, T. and De Koster, R. (2003) An approximation for determining the optimal picking batch size for order picker in single aisle warehouses. In: Meller, R., Ogle, M.K., Peters, B.A., Taylor, G.D., Usher, J. (Eds.), Progress in Material Handling Research: 2002, pp. 267−286.

Le-Duc, T. and De Koster, R. (2005a) Determining the optimal number of zones in a pick-and-pack order picking system. Report ERS-2005-029-LIS, RSM Erasmus University, the Netherlands.

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

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

Lee, M.K. (1992) A storage assignment policy in a man-on-board automated storage/retrieval system. *International Journal of Production Research*, **30**(10), 2281−2292.

Liu, C.M. (1999) Clustering techniques for stock location and order-picking in a distribution center. *Computers & Industrial Engineering*, **26**, 989−1002.

McGinnis, F., Francis, R.L., and White, J.A. (1992) Facility Layout and Location: An Analytical Approach. Prentice-Hall, Egnlewood Cliffs, NJ.

Malmborg, C.J. and Bhaskaran, K. (1987) On the optimality of the cube per order index for conventional warehouses with dual command cycles. *Material Flow*, **4**, 169−175.

Malmborg, C.J. and Bhaskaran, K. (1989) Optimal storage assignment policies for multiaddress warehousing systems. IEEE Transactions on Systems, Man and Cybernetics, **19**(1), 197−204.

Malmborg, C.J. and Bhaskaran, K. (1990) A revised proof of optimality for the cube-per-order index rule for stored item location. *Applied Mathematical Modelling*, 14, 87−95.

Malmborg, C.J. (1995) Optimization of Cubic-per-Order Index layouts with zoning constraints. *International Journal of Production Research*, **33**(2), 465−482.

Malmborg, C.J. (1996) Storage assignment policy tradeoffs. *International Journal of Production Research*, **34**(2), 363−378.

Manzini, R. (2006) Correlated storage assignment in an order picking system. *International Journal of Industrial Engineering*, **13**(4):384-394.

Meller, R.D. and Bozer, Y.A. (1996) A new simulated annealing algorithm for the facility layout problem. *International Journal of Production Research*, **34**(6), 1675-1692.

Meller, R.D. and Parikh, P.J. (2006) Selecting between batch and zone order picking strategies in a distribution center. Technical report, Virginia Tech.

Muralidharan, B., Linn, R.J., and Pandit, R. (1995) Shuffling heuristics for the storage location assignment in an AS/RS. *International Journal of Production Research*, **33**(6), 1661−1672.

Pan, C.-H. and Liu, S.-Y. (1995) A comparative study of order batching algorithms. *Omega International Journal of Management Science*, **23**(6), 691−700

Pandit, R. and Palekar, U.S. (1993) Response time considerations for optimal warehouse layout design. *Journal of Engineering for Industry*, **115**, 322−328.

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

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

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

Petersen, C.G. (2000) An evaluation of order picking policies for mail order companies. *Production and Operations Management*, **9**(4), 319−335.

Petersen, C.G. (2002) Considerations in order picking zone configuration. *International Journal of Operations & Production Management*, **27**(7), 793−805.

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

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

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

Pohl, L.M., Meller, R.D., and Gue, K. R. (2009b) An analysis of dual-command operations in common warehouse designs. *Transportation Research Part E*, **45**, 367-379.

Polito et al. (1980) Solution of spatial equilibrium problem with benders decomposition, *Management Science*, **26**(6) 593-605.

Roll, Y. and Rosenblatt, M.J. (1987) Shifting in warehouses. *Material Flow*, **4**, 147−157.

Ratliff, H. D. and Rosenthal, A. S. (1983) Order-picking in a rectangular warehouse: A solvable case of the travel salesman problem. *Operations Research*, 31(3), 481-501.

Roodbergen, K.J. 2001. Layout and routing methods for warehouses. PhD thesis, RSM Erasmus University, the Netherlands.

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

Roodbergen, K.J. and Vis, I.F.A. (2009) A survey of literature on automated storage and retrieve systems. *European Journal of Operational Research*. **194**, 343-362.

Rosenblatt, M.J. and Eynan, A. (1989) Deriving the optimal boundaries for class-based automatic storage/retrieval systems. *Management Science*, **35**(12), 1519−1524.

Rosenwein, M.B. (1994) An application of cluster analysis to the problem of locating items within a warehouse. *IIE Transactions*, **26**(1), 101−103.

Rosenwein, M.B. (1996) A comparion of heuristics for the problem of batching orders for warehouse selection. *International Journal of Production Research*, **34**(3), 657-664

Sadiq, M., Landers, T.R., and Taylor, G.D. (1996) An assignment algorithm for dynamic picking systems. *IIE Transactions*, **28**(8), 607-616.

Smith, J.S. and Peters, B.A. (2001) Dynamic reslotting for Distribution Center Picking Operation: A Case Study., Technical report, Auburn University.

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

Tompkins, J.A., White, J.A., Bozer, Y.A., Frazelle, E.H., and Tanchoco, J.M.A. (2003) Facilities Planning. John Wiley & Sons, NJ.

Ullman, J. D. (1971) The performance of a memory allocation algorithm. Technical report 100, Princeton University, Princeton, NJ.

Uzsoy, R. (1994) Scheduling a single batch processing machine with non-identical job sizes. *International Journal of Production Research*, *32*, 1615-1635.

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.N., Pochet, Y. (1998) Forward−reserve allocation in a warehouse with unit-load replenishments. *European Journal of Operational Research*, **111**(1), 98−113.

Wascher, G. (2004) Order picking: A survey of planning problems and methods. In: Supply Chain Management and Reverse Logistics, pp. 323−347.

Whilhelm, M.R. and Ward, T.L. (1987) Solving quadratic assignment problems by 'simulated annealing', *IIE Transactions*, **19**(1), 107-119.

Wolsey, L.A. (1998) Integer programming, Wiley & Sons.

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

Yu, M. and De Koster, R. (2008) Performance approximation and design of pick-and-pass order picking systems, *IIE Transactions*, **40**, 1054-1069.

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

Appendix
Correlated Pick Wave Generation

A Correlated Pick Wave Generation

In this appendix, we provide a simple and effective methodology to generate a carton list in a pick wave, in which we can explicitly control the carton size, the total number of SKUs, and the correlation between items in specific cartons for a carton picking warehouse. In the case that a historical order list or carton list with correlation between items is not able to collect from the analyzing warehouse, this methodology is able to generate an effective probabilistically correlated random carton list using predefined correlation matrix or induced correlation matrix with two factors deciding the degree of correlation of SKUs.

A1 Induced Correlation Matrix

If one could not have a real carton list data including the correlation between SKUs in the warehouse, we have to generate a random carton list. In order to include the correlation between SKUs in the random carton list, we propose an effective correlated random carton list generation methodology with SKUs correlation. If we generate a random carton list including the correlation of a certain SKU, two factors (i.e., the number of correlations and the strength of a correlation) should be considered. In other words, we need to consider the following two factors to decide the degree of correlation for each SKU i.

1) How many ordered SKUs are correlated with SKU i?

2) How strongly the SKUs are correlated with SKU i?

We define c_i as the first factor and w as the second factor. To define the degree of SKUs correlation with c_i and w, the *induced correlation matrix* of ordered SKUs is developed to generate carton list. The index value of row and column in the matrix indicates SKUs and the elements in the matrix indicate the correlation probability of between the SKUs in the row and column. In the induced correlation matrix, we basically should provide higher probability to SKUs ordered together than others. The correlation transition probabilities from SKU i to SKU j are defined in Equation (A1):

Parameters:

N	:	Total number of SKUs in the warehouse
c_i	:	Number of correlated items with SKU i
S_i	:	Set of correlated SKUs with SKU i
$\overline{S}_i$	:	Set of non-correlated SKUs with SKU i
w	:	Correlation weight, where $w \geq (N-1)/N$
f_i	:	The portion of correlated SKUs with SKU i

Correlation transition probability from ordered item i to j is described as follows :

$$\Pr(i, j) = \begin{cases} \dfrac{Nw - N + c_i + 1}{Nwc_i}, & j \in S_i \\ 0, & j = i \\ \dfrac{1}{Nw}, & o/w \end{cases} \qquad \text{(A1)}$$

We assume c_i is $N \times f_i$. For example, if we assume the portion of correlated SKUs f_i for all i is 0.1 and the total number of SKUs in the warehouse N has 1000, approximately 100 SKUs are correlated with SKU i. Once the constant c_i and N are selected, we can decide the correlation weights w.

Observation 1 There is no correlation within the SKUs if $w = (N-1)/N$ and there is high correlation between Si and Si if $w = \infty$.

By Equation (A1), there is no correlation between SKUs with probability $1/(N-1)$ if $w = (N-1)/N$ and there is high correlation between correlated SKUs set with probability approximately $1/c_i$ in the correlated sets if $w = \infty$.

Three degrees of correlation w can be selected by imposing even correlation weight to n correlated SKUs and keeping the even gap of the probability difference between the set of the correlated SKUs and the set of the non-correlated SKUs as the total number of SKUs in the warehouse N is large.

Theorem 1 The degree of correlations equally divides in $w = 1, 2,$ and N, given constant c_i for each SKU i as $N \to \infty$.

Proof Let $D\left[S_i, \overline{S}_i\right]$ be the difference of probabilities between the set of correlated SKUs and the set of non-correlated SKUs with SKU i. Then $D\left[S_i, \overline{S}_i\right] = (Nw - N + 1)/Nwc_i$ for any given $0 \le c_i \le N$. By assigning $w = 1, 2,$ and N into the difference, $D\left[S_i, \overline{S}_i\right]$ is described in Equation (A2).

$$D\left[S_i, \overline{S}_i\right] = \begin{cases} \dfrac{1}{Nc_i}, & w = 1 \\ \dfrac{N+1}{2Nc_i}, & w = 2 \\ \dfrac{N^2+1}{N^2 c_i}, & w = N \end{cases} \qquad \text{(A2)}$$

By *l'Hopital's* rules, $D\left[S_i, \overline{S}_i\right]$ is converged by impacting an equal amount of correlation portion to c_i number of correlated SKUs as $N \to \infty$.

$$\lim_{N\to\infty} D\left[S_i, \overline{S}_i\right] = \begin{cases} \lim_{N\to\infty} \dfrac{1}{Nc_i} = 0\cdot\dfrac{1}{c_i}, & w = 1 \\ \lim_{N\to\infty} \dfrac{N+1}{2Nc_i} = \dfrac{1}{2}\cdot\dfrac{1}{c_i}, & w = 2 \\ \lim_{N\to\infty} \dfrac{N^2+1}{N^2 c_i} = \dfrac{1}{1}\cdot\dfrac{1}{c_i}, & w = N \end{cases} \qquad \text{(A3)}$$

Since the amount of correlation portion to $1/c_i$ has equal 1/2 difference of three different w values in Equation (A3), we proved the degree of correlations is equally divided in w = 1, 2 and N given a constant c_i . □

A2 Correlated Pick Wave Generating Algorithm

We provide a probabilistically a set of correlated random carton lists generation methodology using predefined or induced correlation matrix in this paper. The induced correlation matrix gives a different probability in S_i and $\overline{S}_i$ of each ordered SKU i. Thus, the SKUs with higher probability are likely to select more than the SKUs with lower probability. The correlated random carton list generation procedure explains in detail at *Algorithm* A1.

Algorithm A1: Correlated pick wave generation

```
Step1:     Define the number of cartons $C$, the number of line items $L_j$ for
           carton $j$ for $j = 1, \ldots, C,$ and total number of SKUs in the order
           picking system $N$.
Step 2:    if  Correlation probabilities are predefined  than
               Select the correlation probabilities and construct a corresponding
               correlation matrix.
           else
               Define $N \times N$ correlation incidence matrix.
               Select the number correlated SKUs for SKU $i$ of $c_i$ for $i = \{1, \ldots, N\}$
               from the each row $i$ of the $N \times N$ matrix.
               Define the correlation weight $w$.
               Generate corresponding correlated probabilities and construct the
               induced correlation matrix.
           end if
Step 3:    Construct a cumulative induced correlation matrix from the predefined
           correlation matrix or the induced correlation matrix.
Step 4:    Let $m = 1$.
```

Algorithm A1: Correlated pick wave generation (Continue)

```
Step 5:    while $j \le C$ do
               Let $m$ be the index of the first line item in carton $j$.
               if $m$ does not decide for carton $j$ then
                   Randomly generate a integer number $m$ between $(1, N)$.
               end if
               Let $S_i = \phi$ and $\overline{S}_i = \{\text{all SKUs}\}$, for $i = \{1, \ldots, N\}$.
               Let $l = 1$.
               while $l \le L_j$
                   Go to $m$ row to find a next line-item in the cumulative induced
                   correlation matrix.
                   Randomly generate a real number $r$ between $(0,1)$.
                   Select the column index value $n$ which includes
                   $F(n-1) < r \le F(n)$ from the $m$th row of the cumulative
                   induced correlation matrix.
                   if the item $n \in S_i$ then
                       Go back to the second while procedure in Step 5.
                   else
                       Add SKU $n$ into $S_i$ and eliminate the SKU $n$ from $\overline{S}_i$.
                       Change $n$ into $m$.
                   end if
                   Increase $l$ by 1.
               end while
               Increase $j$ by 1.
           end while
```

A3 Example

For the simplicity of generating a set of carton list in a pick wave, we simplified that c_i for SKU $i = \{1, 2, \ldots, N\}$ as a constant c. Let, $c = 3$, $N = 7$, and $L_j = \{3, 2, 3\}$ for $j = 1, 2, 3$ and $w = 2$. The correlated SKUs of SKU i and the number of correlated SKUs with SKU i are shown in Figure A1.

Figure A1 Corresponding correlated SKUs and 0-1 incident correlation matrix

i	c_i	S_i
1	3	2,5,6
2	3	3,4,7
3	3	1,2,6
4	3	3,5,7
5	3	2,4,6
6	3	1,5,7
7	3	2,3,5

$$\rightarrow \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Then the induced correlated matrix is generated by Equation (A1). Based on the 0-1 incident matrix, we can generate the matrix as follows:

$$\begin{pmatrix} 0 & \frac{11}{42} & \frac{1}{14} & \frac{1}{14} & \frac{11}{42} & \frac{11}{42} & \frac{1}{14} \\ \frac{1}{14} & 0 & \frac{11}{42} & \frac{11}{42} & \frac{1}{14} & \frac{1}{14} & \frac{11}{42} \\ \frac{11}{42} & \frac{11}{42} & 0 & \frac{1}{14} & \frac{1}{14} & \frac{11}{42} & \frac{1}{14} \\ \frac{1}{14} & \frac{1}{14} & \frac{11}{42} & 0 & \frac{11}{42} & \frac{1}{14} & \frac{11}{42} \\ \frac{1}{14} & \frac{11}{42} & \frac{1}{14} & \frac{11}{42} & 0 & \frac{11}{42} & \frac{1}{14} \\ \frac{11}{42} & \frac{1}{14} & \frac{1}{14} & \frac{1}{14} & \frac{11}{42} & 0 & \frac{11}{42} \\ \frac{1}{14} & \frac{11}{42} & \frac{11}{42} & \frac{1}{14} & \frac{11}{42} & \frac{1}{14} & 0 \end{pmatrix}$$

Based on above induced correlation matrix, the cumulative induced correlation matrix can be generated by cumulating of the each row from left to right as follows:

$$\begin{pmatrix} 0 & \frac{11}{42} & \frac{14}{42} & \frac{17}{42} & \frac{28}{42} & \frac{39}{42} & 1 \\ \frac{3}{42} & \frac{3}{42} & \frac{14}{42} & \frac{25}{42} & \frac{28}{42} & \frac{31}{14} & 1 \\ \frac{11}{42} & \frac{22}{42} & \frac{22}{42} & \frac{25}{42} & \frac{28}{42} & \frac{39}{42} & 1 \\ \frac{3}{42} & \frac{6}{42} & \frac{17}{42} & \frac{17}{42} & \frac{28}{42} & \frac{31}{42} & 1 \\ \frac{3}{42} & \frac{14}{42} & \frac{17}{42} & \frac{28}{42} & \frac{28}{42} & \frac{39}{42} & 1 \\ \frac{11}{42} & \frac{14}{42} & \frac{17}{42} & \frac{20}{42} & \frac{31}{42} & \frac{31}{42} & 1 \\ \frac{3}{42} & \frac{14}{42} & \frac{25}{42} & \frac{28}{42} & \frac{39}{42} & 1 & 1 \end{pmatrix}$$

Once the cumulative induced correlation matrix is generated, the line-items in the carton list begin to generate. Since we already set the number of line-items for the carton j as $L_j = \{3,2,4\}$ for $j = \{1,2,3\}$, we next randomly generate the first line-item of each carton. Table A1 describes the first line-items in each carton and the random numbers for generating the succeeding line-items in the carton. In the Table A1, the first line-item in carton 1 is generated SKU 2 by randomly choosing 2 between 1 and 7. For the second line-item in carton 1, we start to search the next line-item at the second row of the cumulative induced matrix. We can probabilistically select the one of pivot values of the second row in the matrix by generating a random real number between 0 and 1. Since the selected random number 0.30 for the second line-item is greater than the cumulative probability 0.071 (3/42) at the second column and is less than or equal to 0.333 (14/42) at the third column, we choose the second line-item as SKU 3. Then we change the active searching row into third row. We randomly generate a real number for third line-item in carton 1. The random number 0.86 for the third line-item in carton 1 is greater than the cumulative probability 0.67 at column 5 (28/42) and less than or equal to 0.93 (39/42) at column 6. Thus, the third line-item in carton 1 is selected as SKU 5. Therefore, the line items in carton 1 constructed as SKU 2, 3, and 6.

Table A1 Line-items generation of each carton in a pick wave

Carton	First line-item	Random number	Succeeding line-item
1	2	0.30, 0.86	{3,6}
2	5	0.24	{2}
3	4	0.30, 0.70, 0.35, 0.17	{3,6,1}

Similarly, we can construct line-items in carton 2 and 3 using the cumulative induced correlation matrix. Notice that the third random number 0.35 in row 3 for carton 3 is eliminated for deciding a line item because the corresponding line item SKU 3 for 0.35 is already selected in S_3. Thus, we generate another random number 0.17 for third line item SKU 1 in carton 3. As result, the correlated carton lists in the three cartons are successfully generated in Table A2. This methodology provides a simple procedure to generate a pick wave reflecting SKUs correlation.

Table A2 Correlated Pick wave

Carton	Line-items
1	{2,3,4}
2	{2,5}
3	{1,3,4,6}