RELATION-BASED ITEM SLOTTING

A Thesis presented to the Faculty of the Graduate School

University of Missouri

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science

---

by

Phichet Wutthisirisart

Dr. James S. Noble, Thesis Supervisor

Dr. C. Alec Chang, Thesis Co-supervisor

JULY 2010

The undersigned, appointed by the Dean of the Graduate Faculty, have examined a thesis entitled

RELATION-BASED ITEM SLOTTING

Presented by Phichet Wutthisirisart

A candidate for the degree of Master of Science

And hereby certify that in their opinion it is worthy of acceptance

_____

Dr. James S. Noble, Thesis Advisor

_____

Dr. C. Alec Chang, Thesis Co-advisor

_____

Dr. Youssef G. Saab

ACKNOWLEDGEMENTS

I would like to express my gratitude toward my professors, colleagues, family and friends whose support and guidance has lead me to my graduation. First, I thank my family who always stands by my side and never once walks away while I am going through hard time.

I thank my professors who provided me guidance and trained me so that I can achieve a Master of Science degree. Dr. Noble, I appreciate all of your advices and your patience to answer my questions. Also, I am grateful for your helps that walked me through many hard and difficult situations. Dr. Chang, thank you for your recommendation that gave me an opportunity to work on projects with Dr. Noble, and your advice that I kept me moving toward my graduation. Thank you to Dr. Jang and Dr. Noble who provided me funds so that I could stay focus on achieving the degree.

I would also like to thank my friends who made my years in MU my precious memory and thank to their friendliness that relieved my loneliness I got when firstly arriving in US.

Last but not least, thank you to my mom and dad. You were, you are and always be my great support. Thank you for your unconditional love and faith you have in me. Your son always loves you.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Relation-based Item Slotting

Phichet Wutthisirisart

Dr. James Noble, Thesis Supervisor
Dr. C. Alec Chang, Thesis Co-supervisor

ABSTRACT

The cost of warehouse operations contributes approximately 20% of the total cost in a supply-chain system, and the cost of the picking process can be as high as 65% of the warehousing cost. Therefore, there is an incentive to improve the picking process in order to reduce the overall cost of the supply-chain system. The item assignment policy which allocates items to storage locations is one of the main decision processes that is aimed at lowering the travelling cost, which is normally considered as waste. Existing methods like the Cube-per-Order-Index (COI) and the volume-based strategy focus on the picking of items individually, the fact that multiple items can be picked in one single route is not considered. In this research, the concept of linear placement that has been used in computer science is adopted to create the Minimum Delay Algorithm (MDA). The method includes the relationship among items in order to generate the item storage layout that minimizes the total walking distance of order pickers. MDA was tested against other methods like CLP, COI, and OOS with 30 study cases. The results show that MDA can provide up to a 40% travel distance saving. Finally, MDA is combined with CLP, which is an iterative improvement method, in order to further improve the solution

Chapter 1 Introduction

1.1 Overview of Order-Picking in Warehousing and Logistics Supply Managements

In logistic supply-chain systems, the warehouse operation requires extensive labor time and cost. The warehouse's operational cost is estimated to be around 20% of the entire supply-chain cost (Rene' et al., 2007; Trevino et al., 2008). As one of the warehousing operations, the order picking process that retrieves items from warehouse storage locations to fulfill customer orders is considered as one of the most labor-intensive and costly operations (Rene' et al., 2007; Gu et al., 2007; Trevino et al. 2008). It alone can constitute as much as 50%-75% of the total warehousing cost (Rene' et al., 2007; Peterson et al., 2004a). Therefore, a lack of the proper management in this area not only leads to long lead times in the supply-chain system, but also high warehousing cost and unsatisfactory service. Consequently, in order to improve the performance of the order picking process, there has been extensive research involving lot size selection, storage assignments (item-slotting), routing policies, and optimal (internal) layout design. Despite considering all these aspects, this research focuses on creating a new item storage assignment policy that seeks to minimize a total travelling distance, taken by order pickers in order to retrieve items. According to the existing research discussed in Chapter 2, the existing item slotting strategies only consider characteristics and properties of individual items, but not item interdependences. The lack of item correlations in item slotting policy could lead to long walking distances, especially when multiple items are

picked within a single route (Mantel et al. 2007). The proposed solution addresses this issue, in order to lowering the travel distance.

One of the major objectives when improving the order-picking process is to maximize the turnover rate of an item picker, which could be both human labor and item picking machine, with respect to a number of picked items per unit time (Jewkes et al., 2004; Mantel et al., 2007). This can be viewed as picker utilization, which determines a number of items the picker can retrieve in specific time period. An increase in the turnover rate (i.e. an order could be delivered to customers or production lines quickly) results in high service level and low operational cost. To achieve such goal, the method that minimizes order-picking time becomes one of the most important issues in warehouse optimization.

The order-picking time can be a composition of several sub-process times such as item-searching, item-picking, routing-setup and travelling. Among these sub-processes, the travelling time is often and widely used as performance index for the picking process, since travel time is a direct and inevitable expense that neither adds value nor enhances other processes (Rene' et al., 2007). Rene et al. (2007) mentioned that around 50% of order-picking time is spent on travelling to storage locations. Although the percentage might varying among warehouses and it could be as high as 65% in some warehouses (Peterson and Schmenner, 1999); therefore, travel time is often considered the major concern (Mantel et al., 2007) while trying to improve the picking process as well as overall warehouse performance.

To optimize the travel time, the walking/travel distance in which the order picker has to take in order to retrieve items from storage locations is often considered as primary objective (Caron et al., 2000; Peterson et al., 2004a, b, 2005; Mantel et al., 2007). The correlation between these two aspects clearly happens in positive direction. That is, an increase in travel distance will result in long travel time. Thus, by reducing the walking distance, the travel time is expected to be reduced, leading to a decrease in warehouse operational cost.

According to the research of Peterson and Aase (2004a), there are three main decisions that impact the minimization of travel distance: picking policy, routing policy and item storage assignment policy. The picking policy determines how orders or picked items can be batched in order to be retrieved within a single route. The routing policy sets up how order pickers traverse the storage area, and item storage assignment identifies a location for each item. According to this research, a significant reduction in travel distance can be found by improving the first two policies, but relatively less reduction is yielded by switching between simple routing policies (e.g s-shape) to the optimal routing policy.

In this research, some existing item storage assignment methods and the characteristics of the item storage allocation (slotting) problem are reviewed, observed and analyzed, in order to recognize the properties of the problem which need to be accounted by the proposed solution. In addition, the linear placement problem (LPP) which has been used in designing logic gate arrays inside electronic circuit boards is introduced and applied to the item-slotting problem. The results produced by the proposed solution are compared with the ones generated by the selected LP method.

1.2 Introduction to Item Slotting Strategy

The slotting strategy, sometimes called item storage assignment or storage allocation, aims to allocate stock-keeping-units (SKUs) to storage locations inside a warehouse while trying to minimize total travelling time or distance for order pickers. The slotting/location strategies fond in the literature can be classified into three categories (Peterson et al., 2004a): dedicated storage (policy), random storage (policy), and class-based storage (policy). Dedicated storage binds each item to a specific location, whereas random storage allows items to be placed anywhere. Class-based storage, sometimes referred as a sub-type of the dedicated policy, creates zones each of which is designated for a specific type of products or items that falls into some specific criteria. In practice, dedicated and class-based policies are often applied in highly manual environments, since they provide an objective to maximize the utilization of order pickers while the random method does not address this issue.

Due to the fact that the random policy does not restrict storage locations of any items, the advantage of using this policy is high space utilization, resulting in small size warehouse (Rene et al., 2007; Gu et al., 2007). Also, changing the layout of a picking area is not restricted by positions of storage locations, which are normally bound to an objective function of dedicated and class-based policies. Therefore, to save the travel distance with the random storage assignment policy, optimizing the routing and picking plans are the major focus. The downside of using the random policy is that some popular items might be stored far away from the pick/drop (i.e. known as P/D or I/O) location, resulting in long travel distances.

The class-based storage assignment policy is referred to as hybrid type of the random policy and the dedicated policy (Rene et al., 2007; Gu et al., 2007). Instead of fitting each SKU to a single location or letting them be allocated anywhere inside the picking area, the method classifies the items into groups. Items of each group are placed randomly inside that group. Doing so allows the items to be easily administered and also reduces the travel distance, since each zone location could be located according to the objective function that may try to minimize travel distances between zones or between zone and I/O location.

In contrast, the dedicated policy, binding each item to a storage location basing on some criteria and objective function, allows the company to avoid item shortages, because the space of each product is reserved at its maximum inventory level. However, this could result low space utilization, since every product have different popularity. In addition, according to Rene et al. (2007), it allows item pickers to become familiar with all item locations, since the locations are systematically defined.

There are several methods for dedicated storage assignment. As described in (Peterson et al., 2005), item popularity, item turnover (i.e. considers the total quantity shipped during a given time period), item volume (i.e. referred as the multiplication of demand and item volume), item pick density (i.e. item popularity per one unit of required space by such item), and COI are all popular methods used for the dedicated storage policy. Out of all these methods, the one that has been most extensively studied (Peterson et al., 2004a, b,2005; Mantel et al., 2007; Gu et al., 2007) is Cube-per-Order Index (COI) that was firstly introduced by Heskett (1963). The method considers the picking frequency of each item during a given period of time (i.e. referred as turnover by

the item turnover method) with respect to its space requirements, in order to identify the importance of each item. Later, more important items are placed to the best warehouse location before less important ones.

The downside of using the dedicated storage assignment is its need of intensive information and solid prediction of future orders, in order to generate the storage locations that fully reflect the objective function (e.g minimizing travel time and distance or maximizing the use of labor). Moreover, using this strategy can lead to low space utilization (Rene et al., 2007). In addition, the travel distance might not be minimized when several items are picked in one route, since all the methods mentioned above consider either the popularity or space requirement of each item individually. The list of picked items may consist of both popular and unpopular SKUs. Thus, even though the most popular item has been picked from the best location that is closest to a drop-off location, the picking process still has to continue until all items in the picking list are picked. In other words, each SKU has relationship with each other (Mantel et al., 2007). Ignoring these SKU relationships may results in long walking distance. Consequently, in this research, the method that concerns the item relationship is proposed.

1.3 Introduction to Linear Placement

The concept of linear placement has been used to arrange a number of interconnected circuit gates for very-large-scale-integrated (VLSI) circuits, in order to minimize their remoteness in one dimension placement so that the related gates are placed close to each other. The gate is sometimes referred as a node and its relationship with other nodes is referred as a net (Yamada et al., 1989; Saab and Chen, 1994; Saab,

1996). For example, in case of eight interrelated nodes, their connections or nets are shown as lines connecting each node in Figure 1.1.

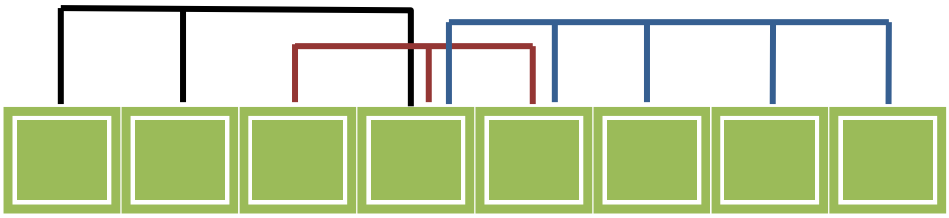

Figure 1.1 A linear placement of eight interrelated nodes with nets connecting each node

The remoteness of these eight nodes is measured by net lengths. That is, the summation of all net lengths represents the overall remoteness of the placement. Thus, the linear placement problem tries to arrange these nodes so that the summation of these lengths is minimized.

Compared to the item storage assignment problem, items or SKUs could be referred as nodes (gates) that are interconnected with each other. In addition, a list of picked items represents a net where all items are picked together. The length of each net represents the travel distance required by a picker to retrieve the items in the list. Therefore, by arranging the items so that their remoteness is minimized, the total travel distance is expected to be minimized as well.

Even though the concept of linear placement has been studied and used for decades in the fields of electrical engineering and computer engineering, it has not been adopted to solve the item storage assignment problem. Trevino (2009) proposed a mixed integer programming to optimally solve the linear placement problem for the warehouse item storage assignment. However, as the size of problems grows, the model's

computational time increases rapidly so that the problem cannot be solved in a practical timeframe.

In this thesis, a method for solving the item storage assignment with the integration of the item relationship is proposed to solve the problem in a practical timeframe and still yield good solution quality compared to COI and order oriented item slotting (OOS) (Mantel et al., 2007). The method generates a linear placement or sequence of items in its first phase, so that the items commonly picked together are placed near each other in linear form. Later, in the second phase, the items are allocated to storage locations according to their orders in the sequence.

1.4 Summary of the Research Objective

The objective of this research is to introduce the concept of item relationships to the item storage assignment problem and to propose an algorithm that applies the concept of linear placement to capture this relationship. To develop the algorithm, current existing methods for the storage assignment and linear placement problems have been reviewed and summarized in Chapter 2. Chapter 3 presents the characteristics of the item storage problem regarding item relationships. To solve the problem, a two phase algorithm is given in Chapter 4: generating an item sequence and generating an item layout according to that sequence. In Chapter 5, the results of both phases are shown while comparing them with the ones generated by the current methods like CLP, COI, and OOS. Finally, the conclusions of this research and as well as related future research topics are given in Chapter 6.

Chapter 2 Literature Review

2.1 Overview of relevant literature

In this chapter, two different areas of literature have been reviewed. First, the current storage assignment polices including the item slotting policies are explored with respect to the order picking process, in order to realize the advantages, disadvantages and improvement possibilities of each method. Second, the linear placement problem literature has been reviewed in order to apply its concept to improve the item storage assignment. Finally, a summary of what has been found and what was selected for developing the proposed method is discussed in section 2.3.

2.2 Item storage assignment and item slotting in the order picking

According to Gu et al. (2007), the order picking process is considered the most expensive warehousing operation since it requires intensive amounts of labor and capital. The major objective of picking process improvement is to minimize the total picking time (Caron et al., 2000; Jewkes et al., 2004) which is typically a function of travel distance (Kallina et al., 1976; Caron et al., 1998). Thus, research has focused on reducing the total travel distance (Peterson et al., 2004a, 2004b, 2005; Mantel et al. 2007; Caron et al. 2000) by either improving the design of the picking area (e.g. numbers of aisles and aisle lengths), or improving operating policies (e.g. picking policy, routing policy, and storage policy). The major focus of this thesis is to improve the storage policy. Therefore, the previous research related to this topic is the focus of this review.

At the operational level of allocating items to storage locations, two processes are commonly included: item storage assignment and item slotting. According to Peterson et

al. (2005), item storage assignment determines how and where to assign each item to a storage location. Meanwhile, the slotting method (i.e. sometimes called a location rule) refers to how items are ranked in order to reflect the objective of assigning the best locations inside the picking area. However, these two terms are sometimes used interchangeably, since the process of assigning items to storage locations consists of both sub-decisions.

Gu, el at., (2007) conducted a comprehensive review on research in warehouse operations. Their work provides a good overview of the area of warehouse-operation design that includes the item storage assignment and slotting problems. The authors classified the storage location assignment problems into three different categories basing on the amount of information known about items and products. The problems based on the product information are commonly used more than the other two types that are based on either empty information or only item information such as arrival and departure time. To solve the problem when the product information is known, the authors defined the three most frequently used criteria for item slotting: Popularity, Maximum Inventory and Cube-per-Order Index (COI). According to their findings, COI was referred to as the most comprehensively studied method. The COI method takes into account both item popularities and the space requirements of each product type. This results in COI working the same way as the item popularity method does when each item requires the same amount of space.

De Koster et al. (2007) provided an extensive review on the literature related to the picking operations inside a warehouse. For item storage assignment, the methods used to solve the problem were classified as either random-based storage assignment policy,

dedicated storage assignment policy or class-based storage assignment policy. While each policy operates differently, they all have their own advantages and disadvantages in term of space utilization and item administrative ability. In addition, the level benefits associated with each policy depend on other warehousing operations such as routing and picking policies.

The advantages of these three methods in term of travel distances was measured and observed by Peterson et al. (2004b). They compared the performance of class-based storage (CBS) assignment to the volume-based (VBS) and random storage assignment policies. In terms of percentage saving, the results obtained by simulation showed that CBS yielded less picking time than the random method. In addition, the percentage saving tended to be low when relatively large pick lists (i.e. containing many SKUs) were used. Moreover, the performance of CBS approached VBS when the number of classes increases.

According to Peterson et al. (2004a), the five most commonly used item slotting policies are: item popularity, turnover, volume, pick density and cube-per-order index (COI). All of these policies consider the demand for each product, but may or may not consider the space requirement of each item. For the item slotting policy, the cube-per-order index has been most extensively studied and used (Gu et al., 2007) due to the fact that it covers both item popularity and space requirements. The method was first introduced by Heskett (1963) and had been discussed by Kallina and Lynn (1976) that it yields an optimal solution when certain assumptions are followed.

Kallina and Lynn (1976) summarized the background of COI used in a distribution warehouse where staging and reserve areas are involved. The basic steps of the COI rule were given in their work, as well as the four common determinants (i.e. compatibility, complementarily, popularity and space) used or avoided in the designing of an item slotting method.

De Koster et al. (2007) mentioned that the amount of travel distance in the picking process depends on many factors such as routing policy and item storage assignment policy. Caron et al. (1998) developed an analytical model to determine the expected travel distance in the warehouse that uses either traversal or return routing policies with the COI as item slotting strategy. To compare both routing policies, a simulation model was used to compare the quality of the results versus those obtained by an analytical model. The results showed that the traversal routing policy outperformed the return policy, except cases where the number of picks per aisle is low.

In order to integrate the issue of item weight into the storage assignment problem, Hwang et al. (2003) sacrificed warehouse throughput in order to obtain an improvement in human safety in the picking process. The authors proposed a linear programming model whose objective function is to minimize the workload of item pickers. In the model, each item's weight, volume and turnover were expressed as a Density-Turnover Index (DTI). Later, the results showed that the COI produced lower travel distances, but had higher energy consumption than DTI.

Jewkes et al. (2004) solved the problems of product location, products-to-pickers allocation, and picker home base location concurrently in a product picking line with

multiple servers. The objective of the integrated problem is to minimize the expected order cycle time; in other words, maximize the expected number of orders that can be completed per unit time. The authors assumed that a picker returns to his/her home base after visiting a bin and then used each bin location's probability of having an order containing at least one product from such location, in order to identify the location of products. Since every picker is expected to return back to his/her home base after each pick, the chance of visiting multiple bins within a single route, as well as item interdependence, were not considered.

To measure the impact of different picking, routing and storage assignment polices, Peterson et al. (2004a) empirically showed that switching from a simple routing policy like traversal routing to an optimal routing resulted in less travel distance savings than changing among picking or item slotting strategies. In their research, the experiment clarified the effect of these three decisions by selecting three different policies for every decision (i.e. routing, picking and slotting), and then generating a number of combinations for these policies. The travel distance resulting from using each combination was compared against each other. According to their results, the within-aisle item storage assignment that applies the concept of volume-based slotting yielded better results than either class-based or random storage methods.

Peterson et al. (2005) investigated the impact of different slotting and storage assigning methods on the order picking process. The five criteria (i.e. sometimes referred as slotting measures) for slotting methods, which are item popularity, turnover, volume, pick density and COI were used to determine a rank for each item in order to identify when item is assigned first to the best location inside the picking area. To create testing

scenarios, each of the selected slotting methods were combined with one of six storage assignment strategies either considering or not considering the use of golden zones (i.e. the area between a picker's waist and shoulders). The results obtained by a Monte Carlo simulation showed that COI yielded the best travel distances among the five slotting schemes. Meanwhile, COI, turnover and item popularity methods provided comparable results to each other in term of total fulfillment time.

Even though several slotting strategies have been developed and deployed, methods like COI and item popularity look at SKUs independently from other SKUs. In other words, the characteristics and properties of each item (e.g space requirement and popularity) are considered separately from other items. Mantel et al. (2007) argued that the lack of considering item interdependence could result in long travel distance, since pickers may retrieve several items in one picking route. To solve this problem, the authors introduced the order oriented slotting (OOS) method that considers the relationship of two items combinations. OOS results in significant distance savings when comparing its results to the popular slotting strategies like random slotting and COI. The mathematical model developed is hard to solve when the number of SKUs grows. The OOS heuristic only considers the relationship between two items, and no characteristics of the pick lists such as length and group of common items are taken into account.

Trevino et al. (2008) proposed a mixed integer linear programming model that ranks items according to their relationships while considering the characteristics of pick lists. However, due to the model's complexity, problems with relatively large size become difficult to solve.

2.2 Linear Placement

The linear placement problem (LPP) seeks to arrange items so that their remoteness is minimize. Remoteness is defined as the distance between related items. For decades, the problem has been used by computer scientists and engineers to allocate circuit gates or chips on very-large-scale-integration (VLSI) circuit board. The components being arranged and their correlation are normally referred as nodes and nets, respectively (Ohtsuki et al., 1979; Saab and Chen, 1994; Saab, 1996). According to Ohtsuki (1979), Yamada (1989) and Saab (1996), the LPP is an NP-hard problem. Thus, a heuristic method is necessary to solve the problem. Due to the fact that the problem considers the correlation among items being arranged, the concept seems suitable to be adopted to solve the item slotting problem presented in this research.

Ohtsuki et al. (1979) developed a heuristic algorithm that converts the LPP into an interval graph. The authors show that the number of tracks connecting gates relates to the chromatic (or clique) number of the graph. The method was tested with seven different small cases (i.e. involving 48 nodes and 48-50 nets) and yielded less than or equal clique numbers obtained by previous approaches that applied a branch and bound algorithm. In addition, as a future improvement the authors suggested that a construction algorithm that can obtain good initial vertex orderings and an iterative improvement scheme could be used to iteratively improve the proposed method.

Kang (1983) created a construction heuristic that selects the most lightly connected node and places them one at a time. The method then folds the one-dimension placement (linear placement) to create a two-dimension placement. Different from Kang's method, Yamada et al. (1989) proposed a hierarchical algorithm that breaks the

original LPP into multiple levels in bottom-up manner, such that the nodes contained by the nets with fewer terminals (or nodes) are assigned to the placement in top-down manner.

Saab and Rao (1991) provided a construction linear ordering (CLO) algorithm that identifies each node's location by considering the number of nets connecting to that node. In addition to CLO, the authors proposed an iterative stochastic algorithm called Stochastic Evolution (SE). The SE iteratively moves items to create a new permutation or placement if the reduction in total wire length satisfies the randomly selected threshold. Both CLO and SE were tested against a simulated annealing algorithm (SA) and Kang's algorithm (Kang, 1983). The results showed that CLO yielded lower total wire lengths for most cases. Meanwhile, SE significantly improved the solutions obtained by CLO and produced significantly better results than the SA.

Saab and Chen (1994) developed an iterative improvement heuristics called limited exhaustive search strategy (LESS). The method repetitively improves the solution obtained in a previous repetition by either moving blocks of nodes and/or reversing the order of nodes inside each block. During each iteration nodes are grouped into several clusters. When there is only one cluster left in the permutation, it is broken down into individual nodes and then the process repeats if improvement was obtained in the previous iteration. However, due to high computational time of LESS, it was modified into the FAST algorithm that considers only small sized clusters. Consequently, the computational time was improved significantly and the results obtained by FAST are close to the best solutions obtained by the literature.

Saab (1996) later improved his algorithm by applying the concept of maximum weight matching to help the clustering process. This algorithm implements his observation that the significant improvement would be obtained if the two farthest nodes of each net are moved closer to each other, and also utilized the concepts of maximum weight matching and node compaction. As a result, the computational time of the new algorithm (i.e. referred as CLP) was improved significantly while its results moved closer to the optimal values.

2.3 Summary of Literature Reviewed

According to the literature reviewed, assigning items to locations required several types of decision such as routing policy, picking policy, storage assignment policy and warehouse's layout design. The storage assignment policy, which is the major focus of this thesis, consists of two sub-decisions that commonly referred as item storage assignment problem: storage assignment and item slotting (Peterson et al., 2005). Three major storage assignment policies mentioned by Peterson et al. (2004a, 2005), Rene et al. (2007) and Gu et al. (2007) are random, dedicated and class-based storages. Even though each of them has its own advantages and disadvantages, the random storage method does not focus on reducing the travel distance. Thus, the method is not expected to be applied with the item slotting approach proposed in this thesis.

In addition to the item storage assignment, the item slotting utilizes several different criteria such as item popularity, maximum inventory, turnover and COI (Peterson et al., 2005; Gu et al., 2007). Out of all existing methods, COI developed by Heskett (1964) has been extensively studied, since it considers both item turnover and space requirement. However, these popular methods considered each item's properties

(i.e. turnover and space requirement) separately from other items. The methods like COI and item popularity were designed with the assumption that a picker would return to his/her home base or I/O location after each pick (Mantel et al., 2007). Thus, they may result in high travel distances if multiple items need to be picked within a single route since the item relationship and the characteristics of the pick list are neglected. Item relationships and the characteristics of pick lists will be discussed further in Chapter 3.

In order to integrate these issues into the item slotting problem, the concept of linear placement that tries to minimize the remoteness of items by placing items with high correlation near each other is worth examining. Originally, the concept of linear placement has been used for the design of VLSI. According to the literature has been reviewed so far, the concept has yet been adopted for the item slotting and storage assignment. In addition, due to the fact that these two problems are different by nature, the concept of linear placement needs to be adjusted to match the proposed problem statement. This modification and the proposed slotting algorithm will be discussed in Chapter 4.

The original problem (i.e. arranging circuit gates) has been found to be a NP-hard problem (Ohtsuki et al., 1979; Yamada et al., 1989; Saab, 1996). Thus, the available methods that could solve the problem in a practical timeframe are heuristic. Saab and Rao (1991) proposed a Stochastic Evolution (SE) algorithm to solve the linear placement problem for the design of VLSI circuit boards. The results showed that the method outperformed Kang's algorithm (Kang, 1983) and a simulated annealing algorithm. Later, Saab and Chen (1994) introduced an algorithm that iteratively improves a solution previously obtained by the algorithm, resulting in significant improvement from the SE

method. However, the computational time of the method grows rapidly when the size of problems grows. Thus, the algorithm has been improved by Saab (1996) by integrating the concept of maximum weight matching and node compaction. According to his results, the algorithm provided significantly reductions in computational time while the solutions

Chapter 3: Problem Characteristic

3.1 Problem Overview

Inside a general warehouse, a picking area contains a number of SKUs, which can range from several hundreds to thousands of SKUs. The items are located on racks that consist of one or more shelves. Each portion of a rack is called bay. A bay can be composed of different shelf levels, which also contain one or more item storage areas, referred as bins (Figure 3.1). Inside each bin there is only one type of SKUs.

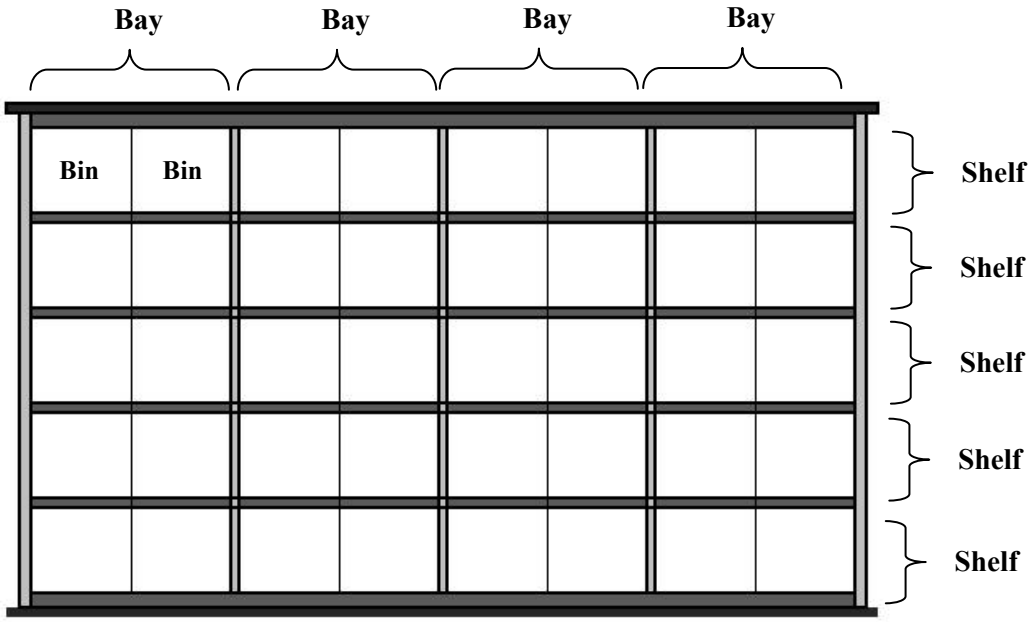

Figure 3.1 A rack with multiple bins, bays and shelves

In this research, the warehouse shape is assumed to be rectangular and filled with identical racks, in term of dimensions and number of bays. In addition, a bay is assumed to have a square top and base with one side access, and contain a single SKU. Each SKU is assumed to entirely occupy one bay and cannot be separated. The proposed storage

assignment policy tries to allocate SKUs to these bay locations in order to reduce the total walking distance required to complete all orders. There is assumed to be one drop-off location per storage area that is located at the bottom left of the area. For ease of walking distance calculation, the width of the walking path is equal to the bay's width. In other words, the entire storage area could be represented as a square grid (Figure 3.2)

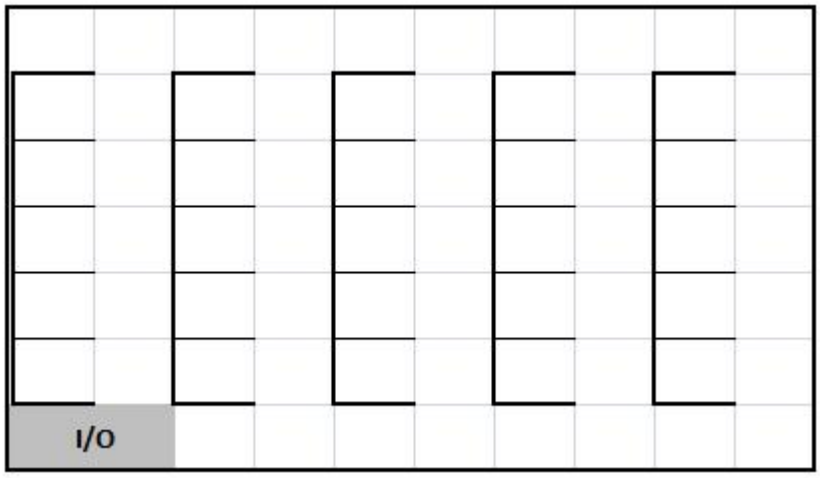

Figure 3.2 A square grid layout of warehouse storage area

Besides the assumption on warehouse layout and storage characteristics, the routing policy adopted by the experiment in this research is an S-shape policy, which requires an order picker to traverse throughout the aisles where items are picked (Figure 3.3).

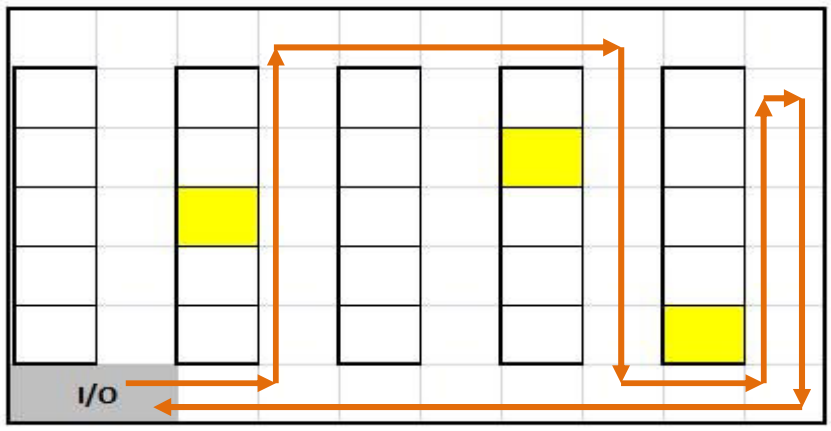

Figure 3.3 S-shape route in the square grid warehouse

Even though the problem is simplified by the above assumptions, due to the simplicity and effectiveness of the proposed method in reducing the total walking distance, the method could easily be adapted to other routing policies and layouts. This will be clarified later when the problem characteristics and factors that affect the performance of the storage assignment are analyzed in the following sections, and when the proposed heuristics are presented in the next chapter.

3.2 List of Picked Items

In one item-picking tour, an order picker retrieves one or multiple items from storage along the route he/she traverses. The items on a list could represent an entire order or part of several orders batched together. Each list could repeat multiple times, and its number of repetitions is referred as order frequency, which depends on the number of orders that contains the same items as the list. In addition, the summation of walking distances produced by each list multiplied by the list's frequency defines the total walking distance for the entire problem.

In this thesis, “order type”, “item list”, “picking list” and “pick list” are used interchangeably when a list of items is mentioned. Furthermore, the words “popularity” and “frequency” refer to the number of times each list repeats.

Besides having its own frequency, each item list consists of a different number of SKUs as shown in Table 3.1.

Table 3.1 An example of order types and their frequency

Frequency	Item List
14	1,9
13	1,2,3,4,7
12	1,3,5,7
10	1,3,7
10	1,3,4,6,8
10	1,8

Minimizing the walking distance required to complete an order depends greatly on the distance between each item in the list. In other words, the ideal distance required to complete a picking process for each order is defined as the sum of bay sizes where all picked items of the list are located. For example, to fulfill an order of three items, the picker has to walk at least three unit distances to reach all three item locations, plus the distance to walk back to the drop-off location (I/O) (Figure 3.4).

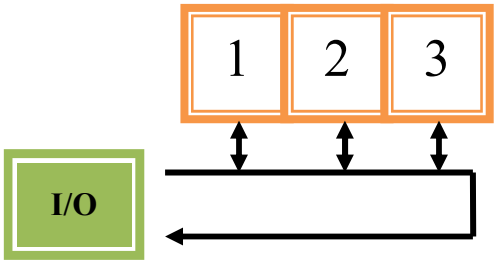

Figure 3.4 The ideal distance for picking three items

However, in real-life cases each order consists of different SKUs. Assigning an item to obtain an ideal walking distance for one order would obviously affect the distances required by other orders belonging to different lists (order types). In this research, an order oriented slotting (OOS) approach which considers the relationship among items is proposed to address this issue.

3.3 Item and Order Popularity

Pick frequency of an item is normally used to determine a location where the item should be located. Items with the highest frequency are regarded as being more popular than ones with lower frequency. Therefore, the high pick frequency items are normally allocated near an I/O location. Probably, one of the most well known and commonly used strategies that applied such a concept is the Cube per Order Index (COI) [Jinxiang 2007]. The COI strategy assigns an item to a location close to a drop-off location according to the ratio between the amount of space required by each item, and the item frequency.

Similar to the individual item popularity, each group of items has its own frequency which determines its popularity against other groups. That is, each list could be executed multiple times. Because of this, the most picked items are not necessary placed closer to the I/O location than the less picked ones. For example, in Figure 3.5, placing items 1, 2, 3 and 4 of a list with X frequency near I/O is preferable over placing them after 5 and 6 that belong to another list having Y frequency, if and only if X is at least twice as large as Y.

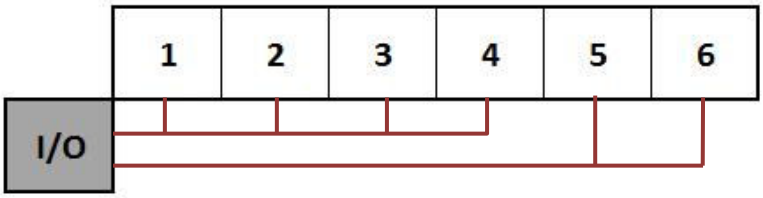

Figure 3.5 A storage layout with six item locations containing six items labeled by numbers

Since there is more than just item popularity (e.g. popularity of item groups) when it comes to item lists that affect the overall walking distance, putting the most picked items closed to the I/O location will not necessarily yield a good final result.

3.4 Location of Less Popular Items

In volume base storage allocation like COI and OOS, items or group of items are normally ranked basing on their picking frequency; then, those with relatively high frequency are stored near the pick-up/drop-off location (I/O) to avoid long walking distances. By doing so, the less picked items are normally placed far from I/O locations. However, as described above, a list of picked items could contain multiple items. Some of the items in a list could be either popular or unpopular. Thus, placing the unpopular items far from I/O location without considering their relationship to other items allows these unpopular items to dominate the walking distance of the entire order.

In fact, volume based methods like COI are not concerned about how each item interacts with another and how often groups of items are picked together. In addition, COI forces unpopular items to be placed far from the I/O. That is, in one picking tour, COI assumes only one item is retrieved and then taken back to the I/O location in each trip. Because of this lack of knowledge of the item relationships, the total walking

distance could be severely affected. In the next section, the effect of item/order relationships will be discussed as one of the important factors contributing in either a low or high total walking distance.

3.5 Item Relationships

As described in the previous section, each pick list consists of multiple SKUs, some groups of SKUs happen to be involved by multiple lists, which have different frequencies. This relationship could be described as item and group of items dependences. For example, in Table 3.2, SKU 2 and 3 appear 12 times in two out of fours order types (lists). Meanwhile, 2 and 4 are picked 15 times and are part of two lists. Thus, these correlations suggest that there is an incentive to place these three items closed to each other.

Table 3.2 An example of order types and their pick frequencies

Order Frequency	Item List
7	1,2,3
5	2,3,4
10	2,4
9	1,3,5

However, COI does not account for this relationship. As result, for the above example it would generate the storage assignment in Figure 3.6. Since the width of the walking path is equal to the bay width for ease of distance calculation, the distance could simply be calculated by counting the number of squares along the walking path. In this example, COI produced 252 units of walking distances.

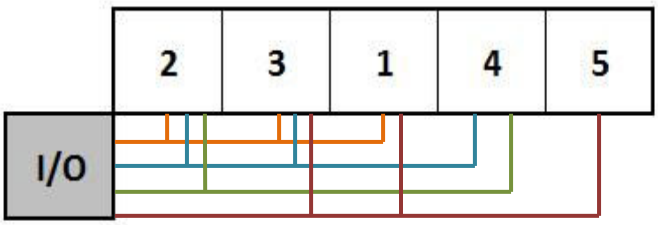

Figure 3.6 Item storage assignment generated by COI according to the data in Table 3.2

In the above assignment, SKU 1 whose popularity is ranked in the third place is assigned in between SKU 3 and 4. This causes a handicap to item pickers in order to complete the second and third order types. In other words, it delays the completion times of the picking process for these two order types by one unit distance. This is in contrast to assigning SKU 2, 3, and 4 closed to each other where the result becomes 36 units less (Figure 3.7).

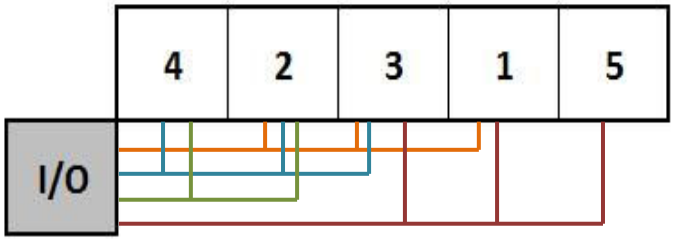

Figure 3.7 Item storage assignment after placing item 2, 3 and 4 close to each other

Even though understanding the relationship among items could lead to an efficient assignment, it is not the only aspect should be considered. Another issue is that the complexity of item relationships grows rapidly as the problem size grows. In the next section, while explaining the importance of list's size, referred as length of item lists, this complexity issue will be illustrated. In addition, during the second phase of the experiment presented in Chapter 4, the minimum delay algorithm (MDA) proposed to

address this issue will be compared with the order oriented slotting (OOS) heuristic invented by R.J. Mantel and P.C. Schuur et al. (2007). According to the literature review, this heuristic is the most current method which concerns the relationships between two SKUs and the distance between their locations.

3.6 Length of item lists

In addition to item popularity and the correlation among items, the length or size of an item list is another issue that should be taken into account while minimizing the total walking distance. The length of a list determines the minimum walking distance in order to complete the picking process for the order. Therefore, minimizing the distance of one order could affect the others composed of items that are not in the list whose walking distance is being minimized.

In the OOS heuristic introduced by R.J. Mantel and P.C. Schuur et al. (2007), this issue is not addressed properly. The effect of such negligence can be seen in the following example (Table 3.3).

Table 3.3 An example of order types and their pick frequencies

Order Frequency	Item List
14	1,9
13	1,2,3,4,7
12	1,3,5,7
10	1,3,7
10	1,3,4,6,8
10	1,8

In this example both COI and OOS generate two similar arrangements (Figure 3.8) with a result of 882 walking units. By considering item relationships, OOS placed

both SKU 1 and SKU 3, which have the highest total frequency among the other item combinations, near the I/O location. However, both methods do not consider the size of the lists containing both items, so assigning SKU 1 and SKU 3 to the storage area closest to the I/O location does not improve the completion time of any lists. Moreover, by doing so, these two SKUs delay the completion time of the picking process of the lists that do not contain either or both SKU 1 and 3, by one and two unit distances, respectively.

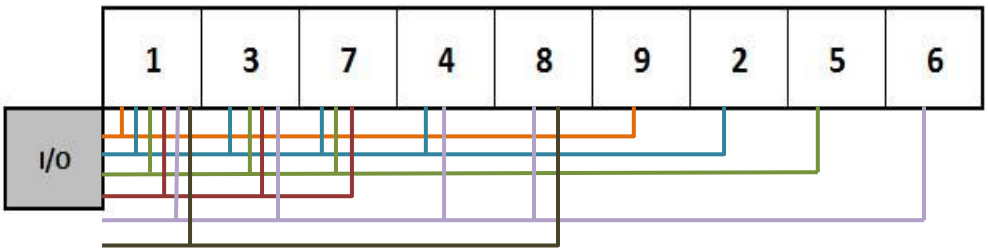

Figure 3.8 Item storage assignment generated by COI or OOS according to the data in Table 3.3

Instead, if the items of the first and last order types are placed near the I/O location, the other lists are only delayed by two walking units contributing by item 8 and 9 (i.e. SKU 1 is a member of all lists so allocating it first does not delay the completion time of any order types). In addition, adding these items allows order pickers to finish the picking process for the first and last order quickly. Consequently, a better layout is obtained (Figure 3.9) with 134 units less distance.

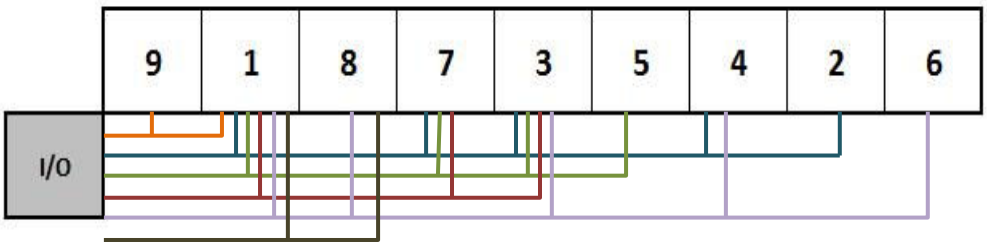

Figure 3.9 Item storage assignment after placing SKU 8 and SKU 9 close to the I/O location

As illustrated by the above example, one of the factors causing COI and OOS to not be able to achieve the optimal walking distance is the negligence in sizes of the orders containing the items selected by both methods. In other words, the lengths of the lists consisting of the selected items could be long enough to diminish the benefit of having them allocated near I/O by elongating the walking distance of other lists.

3.7 S-Shape Routing

The routing policy implemented in this research is assumed to follow a S-shape policy. The policy enforces a picker to traverse through an entire aisle once he/she enters. According to Peterson and Ase et al. (2004), an optimal routing policy provides a significant reduction in walking distance; however the policy is hard to manage and tends to create confusion for order pickers. Thus, many companies prefer simple routing methods like the S-shape over an optimal one. Therefore, an S-shape is selected as the policy to score the item storage arrangements.

Note, the evaluation of the proposed method over a range of batch sizes and routing policies are beyond the scope of this study. These two factors tend to illustrate the importance and effect of item correlation during storage assignment with respect to overall walking distance. However, the reduction caused by the selected layout and S-shape routing is still expected to be present in other combinations of these two aspects. It is empirically shown in [Peterson and Ase et al. 2004] that changes in walking distance shall follow the same trend across the combination of storage assignment, routing policy and batching size.

3.7.1 Walking distance calculation in s-shape policy

The walking distance is a combination of distance in two directions, X and Y (Figure 3.10). In the S-shape policy, the walking distance of one list is determined from the number of aisles containing the items in the list, and the aisle number containing the last picked item of that list. Both numbers indicate the distance in y-direction and x-direction respectively. In addition, since the walking path is assumed to be equal to the bay width, which is assumed to be a square for ease of distance calculation, the walking distance of a list can be calculated as the follows:

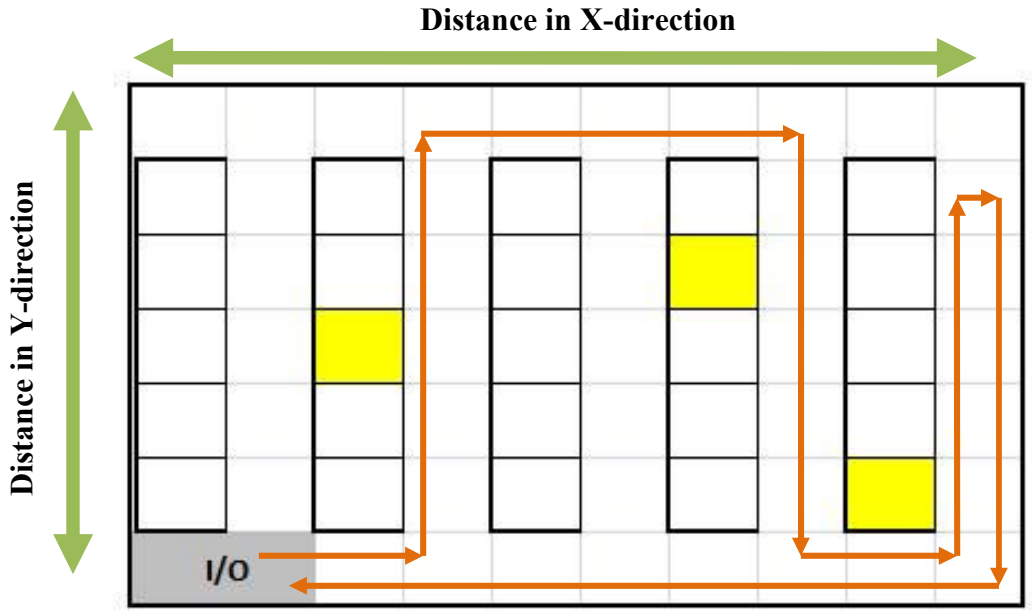

Figure 3.10 S-shape routing in two walking directions (i.e. x-direction and y-direction

Notation:

N	Total number of aisles containing items in the list
M	Aisle size
n_i	The aisle number accessed in ith ordering
d_i	Distance between aisle i and i-1 in X-direction
D	Total distance in Y-direction

X-Direction:

$$Start = 2 * (n_1 - 1)$$

$$d_i = 2 * (n_i - n_{i-1}) + 1 \qquad Where\ i > 1$$

$$Return = \begin{cases} 2 * n_{last} & N\ is\ even \\ 2 * n_{last} - 1 & N\ is\ odd \end{cases}$$

Y-Direction:

$$D = \begin{cases} M * (N + 1) & N\ is\ odd \\ M * N & N\ is\ even \end{cases}$$

$$List\ Walking\ Distance = Start + Return + D + \sum_{i=1}^{last} d_i$$

Chapter 4 Methodology

4.1 Overview of the Methodology

In this thesis, the experiment has been conducted in two phases: linear item sequencing and item allocation. In the first phase, a Minimum Delay Algorithm (MDA) that generates a sequence of items is proposed. This is opposite from COI that arranges the items according to their frequency (i.e. popularity) and required space (Heskett, 1963; Kallina et al., 1976). The sequences generated by the MDA are then compared with ones created by the CLP linear placement algorithm developed by Youssef Saab (Saab, 1996), and by COI (Heskett, 1963; Kallina et al., 1976) and OOS (Mantel et al., 2007).

Originally, CLP was invented to arrange digital gates arrays for very-large-scale-integrated (VLSI) circuit boards, in order to minimize total net lengths connecting the gates. However, due to the fact that the method is used to arrange the gates with respect to the correlation/connection among them, the concept is adopted to rank SKUs, whose correlation is defined by number of times each item appears together. The CLP is an iterative improvement method that requires an initial solution to initialize the algorithm while the linear sequencing method proposed in this research is a construction approach; therefore, both methods are combined in order to gain benefit from each algorithm.

In the second phase, item storage layout is generated based on the item sequence obtained in the first phase. Then, the layouts obtained by MDA are scored against the ones created by COI and OOS.

4.2 Case Study Data

There were two groups of case studies used to test the algorithms. The first group was comprised of 22 cases involving different numbers of items and order types (Table 4.1). The order types contained in these instances were generated through collaboration with Hallmark Cards through the Center for Engineering Logistics and Distribution (CELDi), so that the order types reflect real industrial order types whose items relate with each other (i.e. group of items or SKUs tend to be picked together).

Table 4.1 Statistical information of the 22 study cases

Case No.	Number of items	Number of order types	Avg. Number of items per order	Max Number of items per order	Min Number of items per order	Avg. order freq.	Max order freq.	Min order freq.
1	26	20	2.50	4	2	486.60	984	24
2	44	30	3.00	6	2	595.13	976	40
3	91	50	5.02	21	2	539.44	959	1
4	65	60	2.67	6	2	508.93	951	6
5	138	80	4.48	47	2	457.44	988	17
6	157	100	6.78	29	2	450.43	883	41
7	493	1173	15.16	67	2	249.59	978	2
8	278	1841	19.81	64	1	503.29	1000	2
9	509	3031	23.34	68	2	249.60	978	2
10	486	4164	15.22	66	2	496.35	1000	1
11	512	5797	17.76	67	2	498.75	1000	1
12	524	6500	22.21	64	2	250.54	984	2
13	528	7023	18.70	68	2	497.07	1000	1
14	533	8769	17.23	68	2	500.67	1000	1
15	530	9728	15.65	69	2	497.78	1000	1
16	537	15374	15.90	69	2	496.40	1000	1
17	538	20624	14.83	69	2	502.00	1000	1
18	528	33373	16.95	69	2	501.56	1000	2
19	553	42048	17.87	69	2	502.91	1000	2
20	553	51221	17.76	69	2	502.15	1000	2
21	557	60257	17.13	69	2	501.01	1000	2
22	565	94079	17.17	69	2	500.82	1000	2

The second group was based on the literature (Saab, 1996) and is comprised of 8 cases (Table 4.2), where the frequency of each order type equals one. Even though all frequencies are set to be the same (i.e. since they were used for different type of problems – VLSI design) the correlation among items still exists throughout the lists of order types. Thus, they can be used to determine how well each algorithm addresses the item correlation issue.

Table 4.2 Statistical information of 8 study cases from literature (Saab, 1996)

Case No.	Number of items	Number of order types	Avg. Number of items per order	Max Number of items per order	Min Number of items per order
1	60	75	5.96	11	2
2	100	125	6.35	11	2
3	199	239	6.19	11	2
4	400	421	6.38	11	2
5	600	680	6.43	11	2
6	800	841	6.40	11	2
7	1000	1250	6.50	11	2
8	1497	1610	6.56	11	2

4.3 Phase1: Linear Sequence Generation

4.3.1 Overview of linear sequence generation

Linear sequencing or the linear placement problem tries to minimize remoteness of items or nodes while placing them into a linear form. It has been adopted by electrical engineering, computer engineering and computer science, which uses it to determine the locations of electronic components (gates) in a circuit board. Before assigning the gates to the board, the method arranges them into a linear form where related gates are placed

closely together (Kang, 1983). This is in contrast to the item storage assignment problem where each order picker retrieves an item one after another; the linear sequencing is then applied to order warehouse items so that related items are placed closely together. Thus, item pickers can finish the picking process quickly once they reach the first picked item. In this case, remoteness refers to the total walking distance in the problem.

To determine the optimal linear placement for item storage, a mixed integer programming model was developed by Trevino (2008). However, as the size of the problem grows, the computational time increases rapidly. Therefore, due to the fact that the linear placement problem is NP-hard (Tatsuo et al., 1979; Yamada et al., 1989; Saab, 1996), a heuristic needs to be developed in order to provide a good feasible solution in a practical timeframe. In this study, the Minimum Delay Algorithm (MDA) is proposed to linearly sequence SKUs. The results obtained from this algorithm are later compared and combined with the linear placement CLP algorithm developed by Saab (1996). Finally, these results are scored against existing item storage assigning algorithms' results such as the Cube-per-Order Index (COI) and Order Oriented Slotting (OOS) (Mantel, 2007).

4.3.1 Linear sequencing algorithm CLP

The CLP algorithm that minimizes the remoteness of components arranged in a linear form was developed by Saab (1996) to arrange circuit gates in VLSI design. The method utilizes an iterative improvement algorithm that repetitively improves previously achieved solutions by performing three kinds of operations (i.e. MOVE, FLIP and COMPACT) on each node, as described in section 2.3. The method iteratively packs nodes until one node is left, and then decomposes it back into individual items. Then, the

process repeats these three operations again until no further improvement is obtained and no compaction is achieved.

The primary difference between the linear placement in the VLSI deigning problem where gates can be moved freely to minimize their distance (remoteness) from other related gates and the linear placement in the item storage assignment problem is that item-pickers normally start and end their route at a single I/O location (Figure 4.1 and 4.2). That is, all locations relate to the I/O location, which cannot be moved. Consequently, by adding this constraint, the number of ways to arrange n items increases from $\frac{n!}{2}$ to n! In order to adapt the CLP algorithm to generate an item sequence for the item storage assignment, MOVE considers the I/O location as one of the extreme nodes for every order (i.e. first and last visited nodes), and cannot be moved. Then, the method tries to move nodes close to this location, as well as to the other extreme nodes.

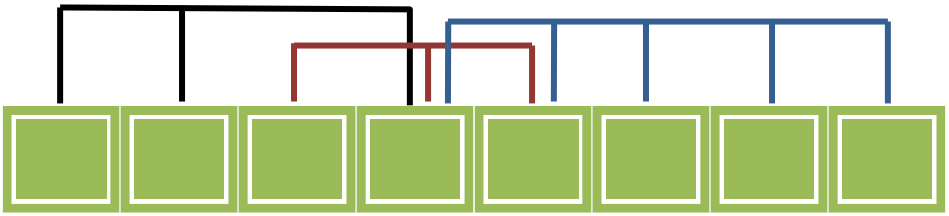

Figure 4.1 Linear placement in the circuit gate assignment problem

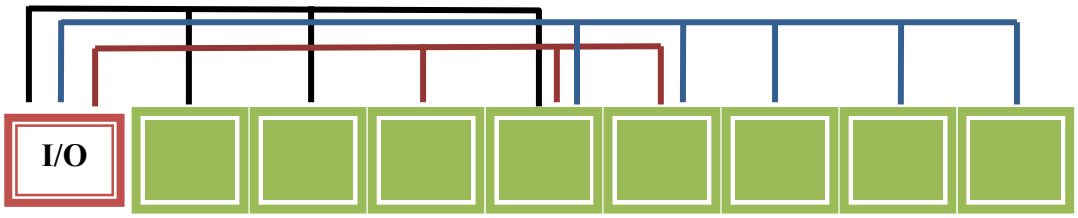

Figure 4.2 Linear placement in the item storage assignment problem

Besides having one end of every order type bound to the I/O location, in the storage assigning problem, each order type or net has its own picking frequency. Thus, while the maximum weight matching is calculated for each pair of consecutive nodes, frequencies of the nets containing both nodes represent their weight. That is, instead of adding one to the total weight, the frequency number is added.

The initial solutions used as input of CLP are randomly generated. Because of this, each study case was run five times and the average, maximum and minimum scores (i.e. total travel distance) and processing time among these five solutions are compared with the solutions generated by MDA.

4.3.2 Minimum Delay Algorithm

Due to the iterative improvement nature of the CLP algorithm, CLP can require significant computational time to solve large scale problems that include more than a thousand SKUs and order types. An approach termed Minimum Delay Algorithm (MDA) was developed while CLP was used as benchmark to compare the quality of results yielded by MDA.

The basic concept of MDA is to assign the current worst item to the current worst location one at a time. The worst item is defined by the total amount of delay in completion time of the order types whose item is assigned the available location that is furthest from the I/O. In other words, the sequence represents orders of items that will be placed into the layout accordingly to the amount of delay each item would cause to the order types that do not contain the item. Once an item is assigned to the worst location,

the order types containing the item are excluded from the calculation of MDA when the item selection process repeats in order to search for the next item to be allocated.

Table 4.3 Notation

N	Set of all items (SKUs)
P	Power set of N
L,X,Y	Item set variables
S	Set of order types (i.e. $S \subseteq P$)
S_i	Subset of S that contains items i (i.e. $S_i \subseteq S \subseteq P$)
W_k	Frequency (weight) of order type k

In addition to the above notations, the symbol | | represents size or amount of required space for an item or item set. For example, if order type k contains SKU 1, 2 and 3 (i.e. k = {1, 2, 3}), where SKU 1 and 2 require one space unit each (i.e. |SKU-1| equals to one, as well as |SKU-2|), and SKU 3 requires two space units (i.e. |SKU-3| equals to two), then |k| is equal to four units of space.

The symbol || || represents the cardinality of a set (i.e. number of elements in the set). However, due to the assumption stating that each item requires only one bay in order to simplify the problem, the size of an item is always equal to one, and also the size of an item set is always equal to the number of items contained in the set. In other words, |Set A| and |SKU| are equal to ||Set A|| and one, respectively. Yet, in order to make the algorithm reflect other cases where the amount of space required by each item is different, this concept and notation are included in the proposed algorithm as well.

The MDA steps are shown below:

1) Set r = 0 and X = N

2) Set d = 0 and Y = {Ø}

3) Set m = |N| - r - d

4) IF $m < 1$ THEN: STOP; Otherwise, move to the next step.

5) Find $set\ L = \left\{ Min_{i\ \epsilon\ X \backslash Y} \left\{ \sum_{k \epsilon\ S_i : |k| \leq m} W_k * (m - |i| + 1 - |k|) \right\} \right\}$

5.1) IF $\left(\left||L|\right| > 1\right)\ AND\ (m > 1)$ THEN:

$d = d + 1$

$Y = X - L$

GO TO STEP 3

5.2) ELSE: r = r + |L|

6) $X = X - L$

7) $S = S - S_i\ \ \forall_{i \epsilon L}$

7.1) IF $S = \{Ø\}$ THEN: $L = X$;

7.2) ELSE: move to the next step.

8) Place all item $i\ \epsilon\ L$ at the location m first and there-before

9) GO TO STEP 2

The first and second steps set up the initial values for “r” and “d” variables and are used to identify a current worst location “m”. In addition, X represents a current set of available SKUs that need to be assigned into a linear permutation (i.e. sequence) and Y represents a set of SKUs that do not need to be assigned into the permutation. They are initialized as a set of all items (i.e. set N) and empty set, respectively. The third step identifies the current worst location which needs to have an item assigned to it. Step 4

checks whether there is an available location left without item assigned to it. If not, the algorithm stops.

In the fifth step, which is the heart of the algorithm, each unassigned item is assumed to be placed in the current worst location m. Then, the amount of delay caused by assigning each item to that location is calculated. The delay of an order is basically the number of additional walking units spent to reach the location of the last item picked in the order. For example, according to Figure 4.3, in order to finish the order consisting of SKU 1, 2, 3 and 4, two extra walking units are required if either one of these four SKUs is placed in location m. After the delay amount is obtained for each item, the ones producing the least amount of delay is selected.

Figure 4.3 Linear permutation with unassigned item (i.e. represented as blank), assigned items (i.e. represented with a number inside) and the current worst available location (i.e. marked by "m")

Figure 4.4 Linear permutation with SKU-4 assigned to the old "m" location.

In case that several items are picked in the fifth step, the worst location is set one location forward of the I/O location. Then, those selected items are compared against each other by again calculating the total delay each of them will produce if it is assigned to this new location. This item selection process in step 5.1 repeats until there is only one item left in Set L or the current available location is the one closest to I/O (i.e. “m” equals

to one). Step 6 excludes the selected items contained in L from the set of available SKUs (i.e. set X).

In step 7, the order types whose items are contained in set L are excluded from set S, which contains the order types that none of their items has been placed in a linear placement. In step 8, the selected items in set L are placed in the worst locations. The process continues with the current assignment location being shifted by |L| units toward the I/O location (Figure 4.4). The algorithm repeats until all items are assigned to a storage location.

4.3.3 Combination of MDA and CLP

Due to the nature of the CLP algorithm that iteratively improves an initial solution the method is expected to produce a sequence whose travel distance is better than, or at least equal to, the initial solution. However, because of this approach the algorithm can spend a significant amount of computational time to solve large problems. This is in contrast to MDA which is a construction algorithm that tend to require less computational time. Thus, in order to gain the advantage of both algorihtms, CLP and MDA are combined. In other words, MDA is used to construct the initial solution that is used as the initial solution for CLP in order to achieve an even better result and to reduce overall solution time. According to Saab (1994, 1996) the speed of the CLP algorithm is expected to improve if a good initial solution is used.

Finally, the linear permutations obtained by these three methods (i.e. CLP, MDA, and MDA+CLP) are compared with solutions generated by Cube-per-Order Index (COI) developed by Heskett (1963) and Order Oriented Slotting (OOS) developed by Mantel et

al. (2007). Even though they are not linear placement algorithms, their concept of allocating the most important items close to the I/O location can be applied to create an item sequence. Generally speaking, COI considers the importance of an item based on its popularity and required space. Meanwhile, the OOS heuristic considers the popularity of item pairs. Thus, COI which is probably one of the most popular item storage assignment policies and has been studied most (Trevino et al., 2008; Peterson et al., 2005; Gu et al., 2007) is selected in this research to represent the case where an item's popularity is considered individually without concern for correlation among items. Furthermore, OOS represents the case where item correlation is consider, but the length of each order type is neglected.

4.4 Phase 2: layout generating

In the second phase, a storage layout is created based on the sequence generated by MDA in phase 1. Each item is selected accordingly to its order in the sequence and then placed in the current best available location that is closest to the most recent allocated item in order to maintain item relationship.

According to the assumptions on routing method used in this research, the best locations in the layout are all bay locations in the first column of the layout (Figure 4.5). The next best locations are in the second column, then the third column, and so on. For example, in case of having a sequence of SKUs 1 to 25 as 1 to 25, the layout is generated as shown in Figure 4.5 Note that for S-shape routing a picker is forced to traverse the aisle once he/she enters, so placing SKU 6 at location 10 as shown in the figure is the same as placing it in location 1.

Once a solution is obtained by MDA, the layout is scored against the solutions generated by COI and OOS. In addition, to determining the quality of the sequence for different sizes of racks, a sensitivity analysis was performed by generating layouts when the rack size is equal to 10 bays, 30 bays and 50 bays.

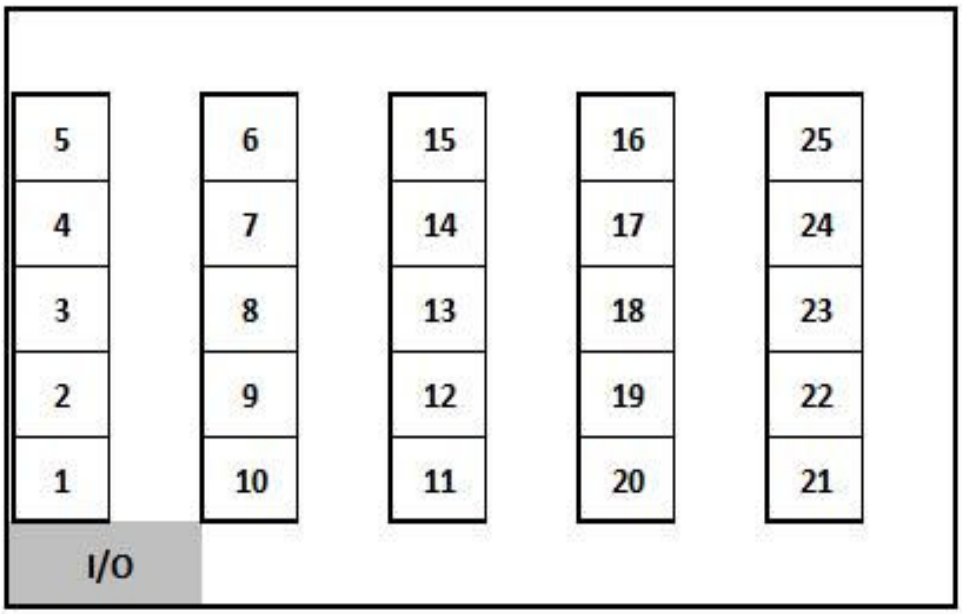

Figure 4.5 Layout with items assigned according to MDA

Chapter 5 Result and Discussion

5.1 Chapter Overview

This chapter presents the results generated by the MDA, CLP, COI and OOS algorithms. Since the experiment has been conducted in two phases, the results of each phase were presented separately. As mentioned in previous chapters, travel distance is a key measure in this thesis to assess the performance of MDA. Also, in order to keep the thesis concise, the travel distance results for each algorithm are presented as the percentage difference comparing the travel distances yielded by MDA against the results of the other methods. In other words, the percentage represents the improvement provided by MDA over the results of the other methods. Thus, a negative percentage refers to the case where MDA alone produces better results (i.e. lower travel distance) than the methods compared against. Detailed results are also given in Appendix A for future analysis.

In addition to measuring travel distances, the computational time of MDA, CLP and the combined algorithm (i.e. MDA+CLP) are empirically captured and compared among each other in the first phase in order to compare the speed of each method.

Due to the fact that the travel distances of the linear permutations generated by MDA and CLP are not significantly different, only the sequences created by MDA and the combined algorithm are used to draw the item storage layout in the second phase. Then, the layouts are compared with the ones created by COI, which is the method that has been studied extensively (Gu et al., 2007), and OOS that includes relationship between two items (Mantel et al., 2007).

5.2 Phase 1: Linear Sequence Generation

5.2.1 Overview

In this phase, the experiment was conducted on MDA, CLP, MDA+CLP, COI and OOS to generate item sequences. The linear travel distance produced by each method observed over 30 study cases was measured. Since MDA is the proposed method, the results are shown as percentage differences calculated as the following equation when X is the method compared against MDA.

$$Percentage\ Difference = \frac{Distance\ of\ MDA - Distance\ of\ method\ X}{Distance\ of\ method\ X} \times 100$$

Therefore, the percentage difference refers to the (percentage) improvement provided by MDA. An exception was made when comparing MDA and MDA+CLP. Since the combined method tries to improve the solution yielded by MDA, the denominator was set as the results of MDA. In addition to measuring distances, the computational speed of each linear placement algorithm (i.e. MDA, CLP and MDA+CLP) was empirically captured. Note that in case when the computational time is less than a second, the time is presented as zero.

5.2.2 MDA versus CLP

This section compares both the travel distance of the linear permutations generated by MDA and CLP separately, and the computational times of both methods. Since the initial permutations fed into CLP are randomly generated, the method was run five times for each study case, in order to obtain average, maximum and minimum results. For each case, the minimum distance obtained and the average computational-time among five runs were used to compare with MDA.

For the first group of data, consisting of 22 study cases and referred as Group 1, the percentage differences in travel distance are shown in Figure 5.1. According to these results, CLP produced around 0.6% lower distance than MDA on average. For the other 8 cases (i.e. referred as Group 2), CLP also produced around 3.29% lower distance than MDA on average (Figure 5.2).

The computational times spent by both methods are graphed against each other in Figure 5.3 and 5.4. The computational time of CLP increases with a higher rate than MDA as the size of problems grows (i.e. number of items or order types increases).

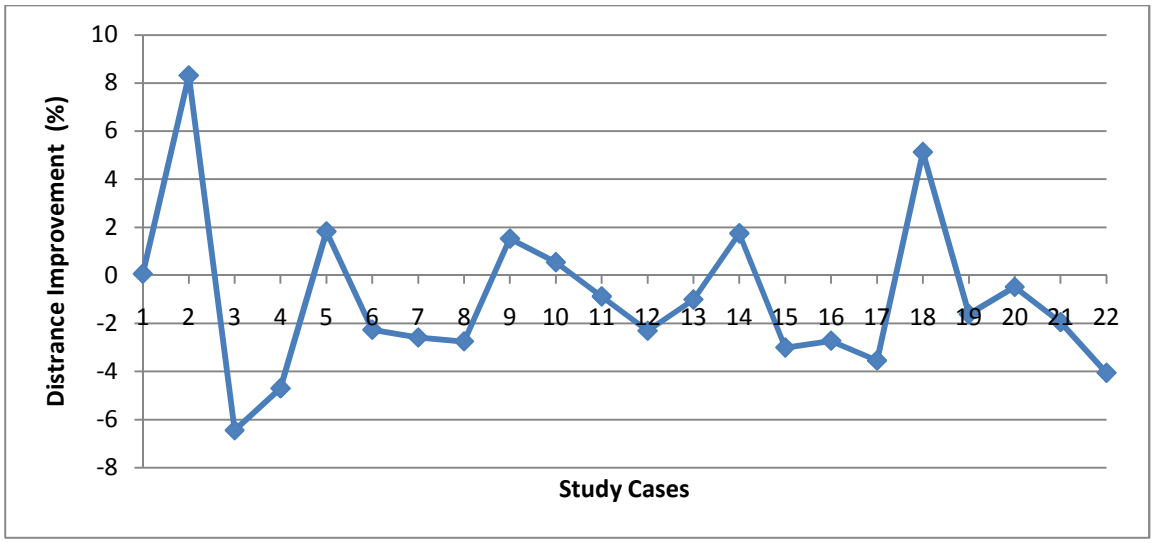

Figure 5.1 Percentage difference in travel distance between MDA and CLP for the dataset 1

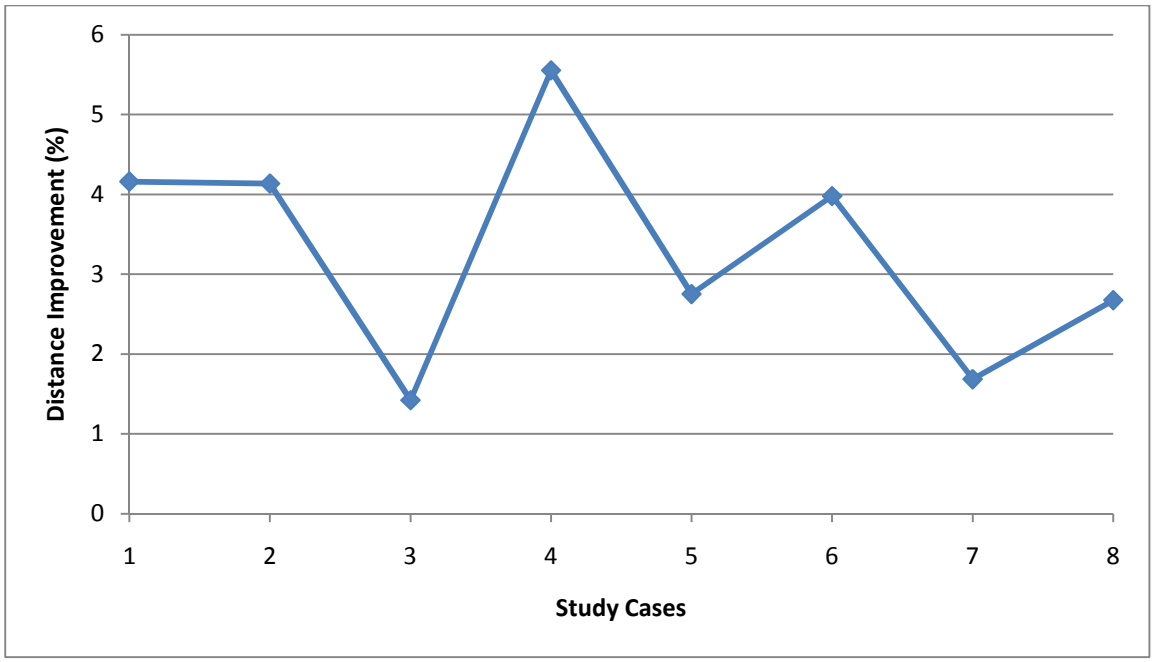

Figure 5.2 Percentage difference in travel distance between MDA and CLP for the dataset 2

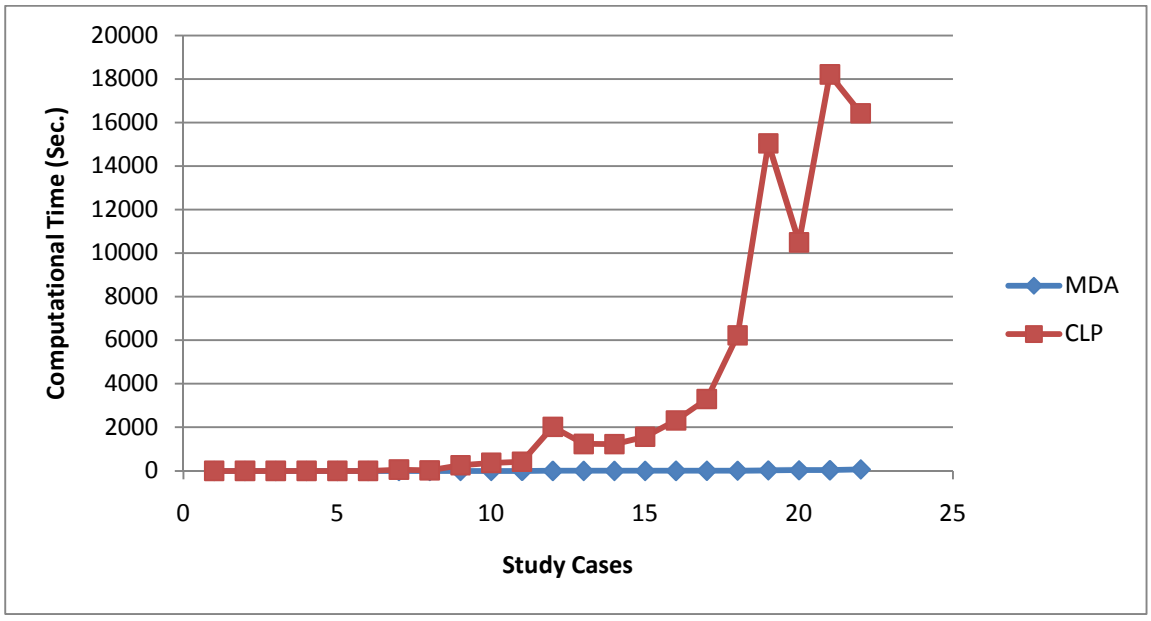

Figure 5.3 Computational times of MDA and CLP on Dataset 1

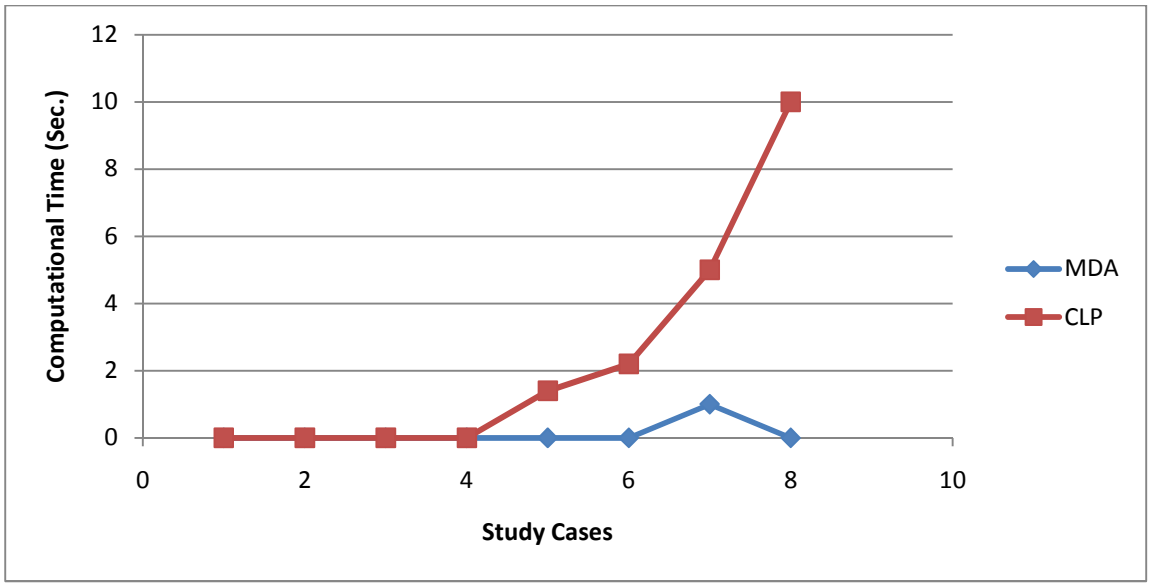

Figure 5.4 Computational times of MDA and CLP on Dataset 2

5.2.3 MDA versus MDA + CLP

Since CLP iteratively improves an initial solution, the method is guaranteed to yield results that are better or at least equal to the initial solution. Therefore, the percentage difference of travel distances is always positive (i.e. the negative percentage refers to the case where MDA produces better results than what has been done by the other method). Meanwhile, comparing to CLP, MDA constructs relatively good solutions in relatively less time.

To obtain the advantage of both methods, CLP and MDA were combined and is referred as MDA+CLP. The combined percentage improvements in travel distances (i.e. improved from the solution generated by MDA) are shown in Figures 5.5 and 5.6. The average percentage improvements of MDA+CLP are around 2% and 4% for the first (i.e. 22 cases) and the second datasets (i.e. 8 cases), respectively.

In term of speed, the computational times of CLP, MDA, and MDA+CLP were shown in Figures 5.7 and 5.8. It was found that the combined method tends to run 34% and 19% on average faster than CLP did for the first and second datasets, respectively. However, they are still much slower than MDA alone.

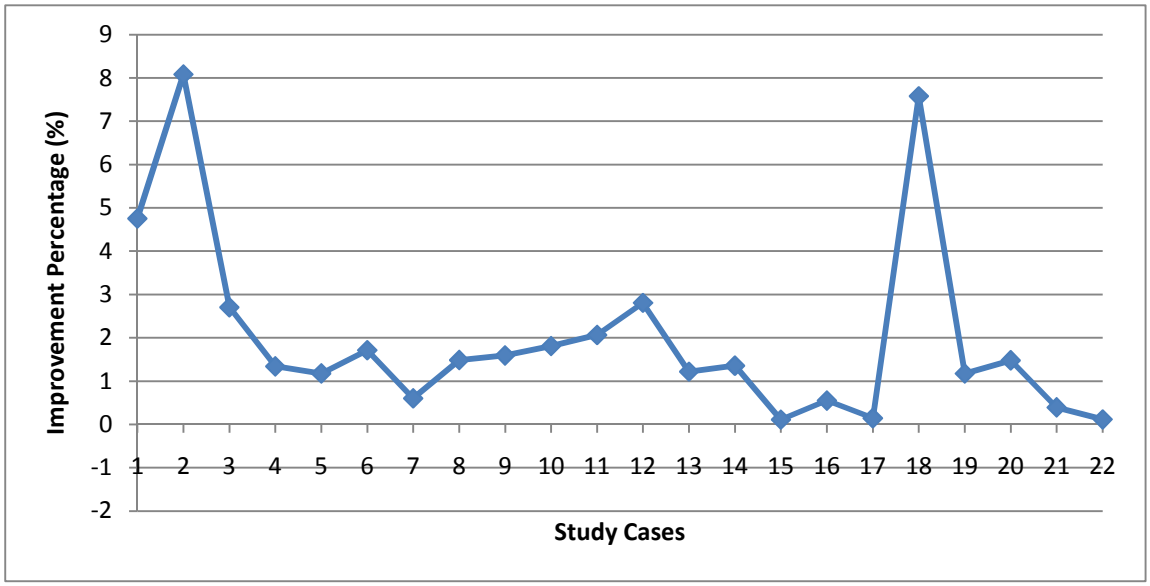

Figure 5.5 Percentage difference in travel distance between MDA and MDA+CLP for Dataset 1

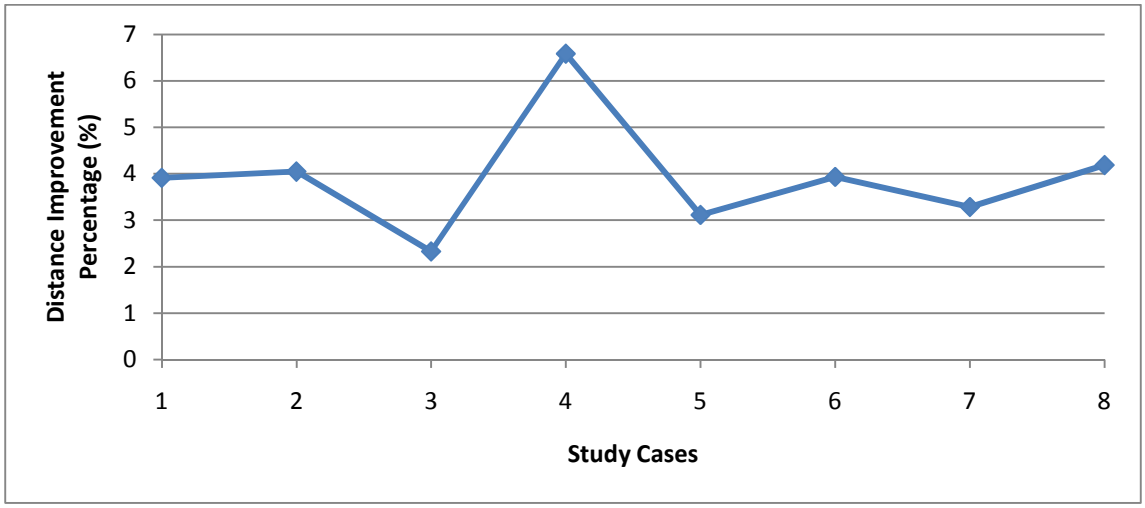

Figure 5.6 Percentage difference in travel distance between MDA and MDA+CLP for Dataset 2

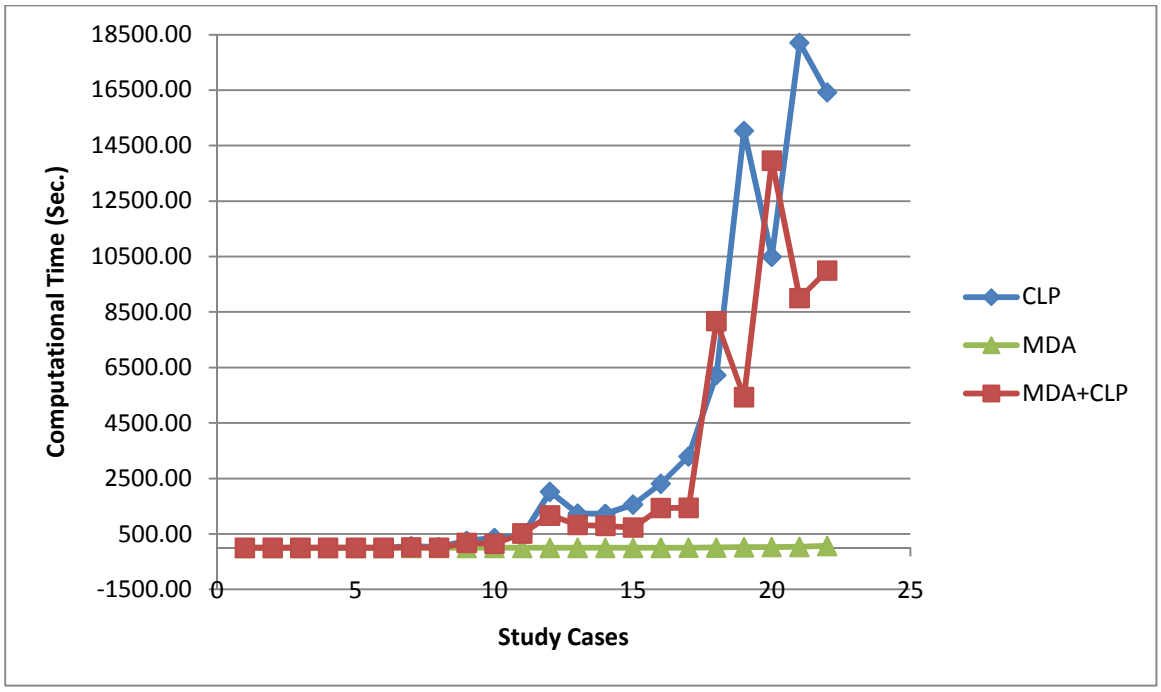

Figure 5.7 Computational times of CLP, MDA and MDA+CLP on Dataset 1

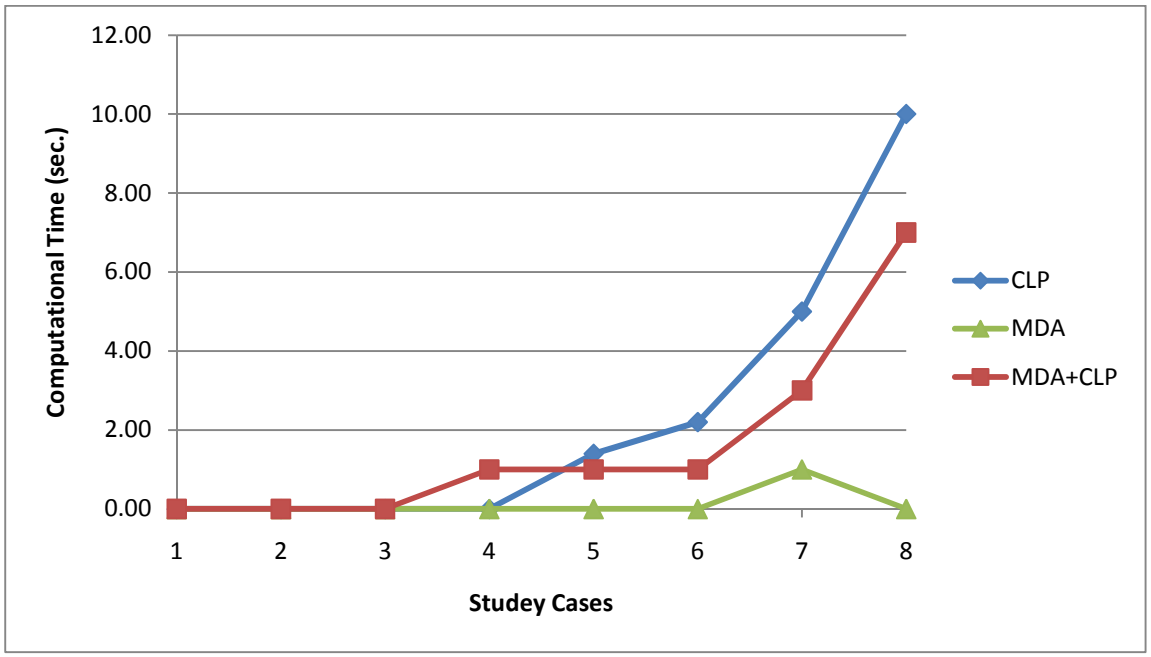

Figure 5.8 Computational time of CLP, MDA and MDA+CLP on Dataset 2

5.2.4 MDA versus COI and OOS

Even though COI and OOS are not linear placement algorithms by nature, they are methods that place items according to the item's importance and can be easily applied to generate a linear sequence of items as well. The item's importance refers to the concept of using the ratio of an item's required space and its picking frequency (Heskett, 1963), or the concept of using the interaction frequency of possible combinations for two items (Mantel, et al., 2007).

The improvement percentage in travel distance when comparing COI and OOS against MDA are shown in Figure 5.9 for the dataset 1 and Figure 5.10 for the dataset 2. Again, the negative percentage implies that MDA produced better results than the other methods.

The results for the dataset 1 show MDA yielded 19% and 21% on average lower distance than COI and OOS, respectively. In addition, it provided approximately 17% better distances for the dataset 2. In some study cases the individual percentage was up to 30%-35%.

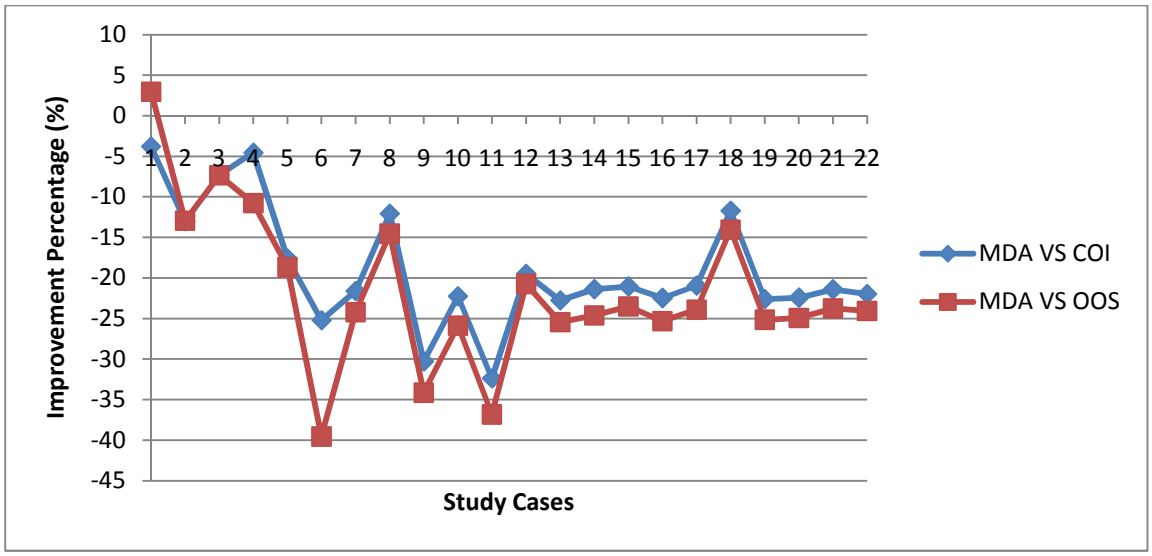

Figure 5.9 Improvement percentages of MDA comparing against COI and MDA comparing against OOS on Dataset 1

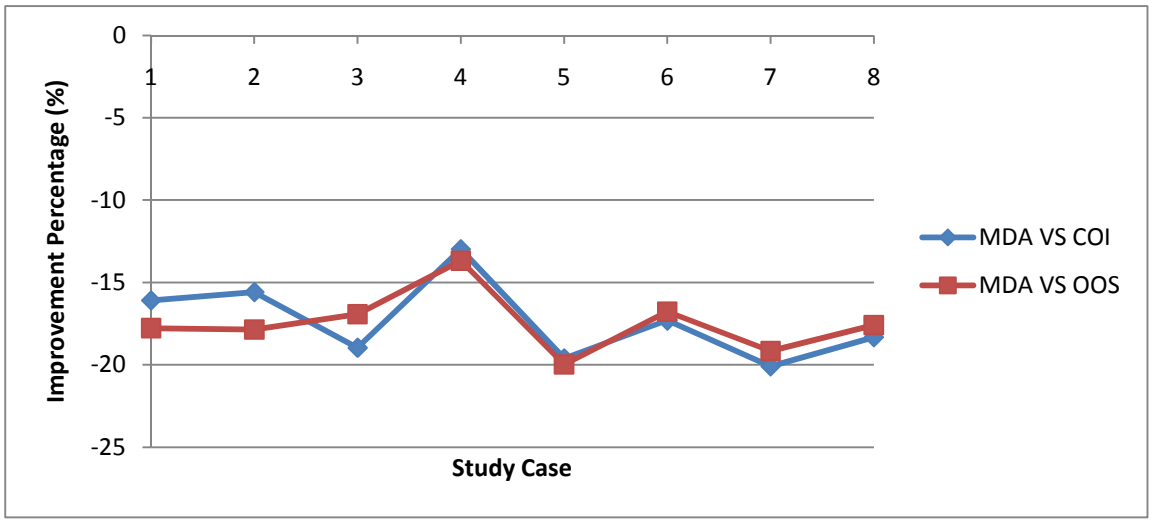

Figure 5.10 Improvement percentages of MDA comparing against COI and MDA comparing against OOS on Dataset 2

5.2.5 MDA+CLP versus COI and OOS

The distances generated by MDA+CLP were compared against COI and OOS. The percentages are plotted in Figures 5.11 and 5.12 for the dataset 1 and 2, respectively. As shown in the Figures, the results of MDA were improved by CLP. In this scenario all results are in the negative Y region, which means that MDA+CLP produced better results than COI and OOS. Also, for both datasets, the average percentage improvements of the combined algorithm over COI and OOS are around 20% - 23%.

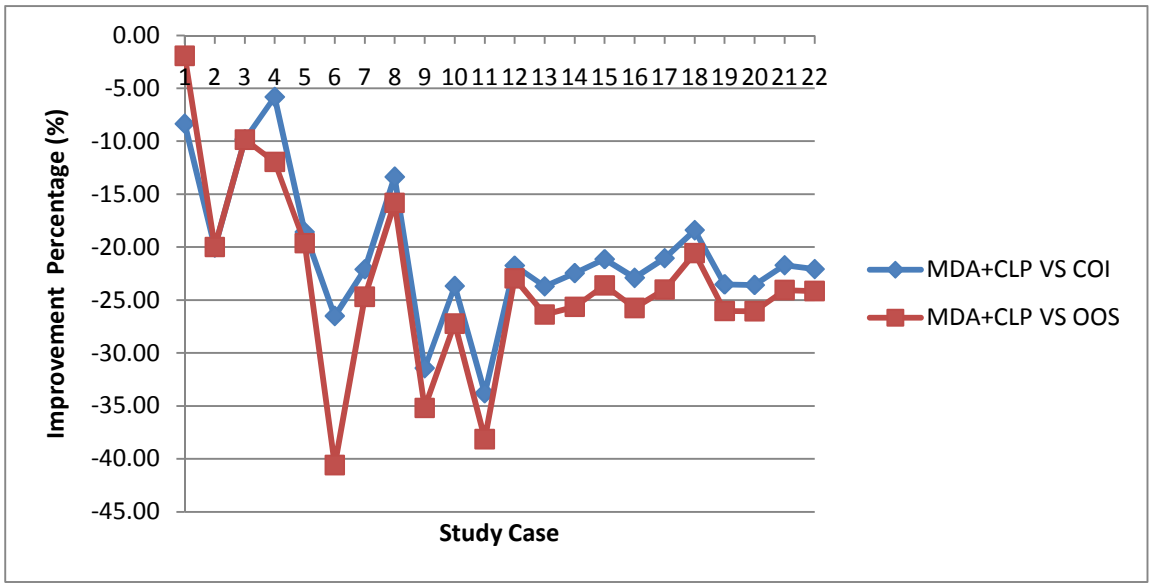

Figure 5.11 Percentage improvements of MDA+CLP over COI and OOS for Dataset 1

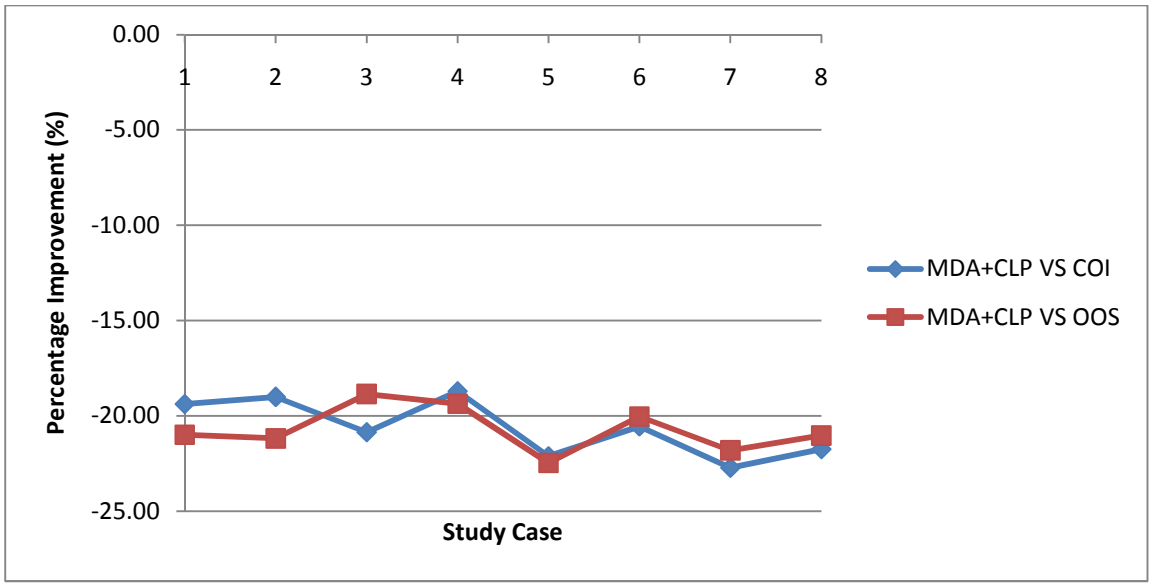

Figure 5.12 Percentage improvements of MDA+CLP over COI and OOS for Dataset 2

5.2.6 Cross result discussion

According to the results shown in sections 5.2.2 to 5.2.5, CLP produced better linear sequences than MDA did for some cases, especially those belonging in the dataset 2. However, compared to CLP, MDA was able to yield comparable or even better results in some cases with much less computational time. In other words, the computational time of CLP increases rapidly when the size of problems grows; meanwhile, the computational time of MDA increases at a much slower rate.

To obtain even better results and also improve the speed of CLP, the combination of MDA and CLP was tested. The combined method is able to yield lower travel distances than MDA, as well as improves the computational time of CLP by 34% and 19% on average for the dataset 1 and 2, respectively. In addition, for some study cases, the computational time was reduced by 75% and 54% for the combined method.

Furthermore, the difference in computational time between CLP and MDA+CLP seems to be more visible in relatively large size problems than in small size problems. According to the computation times spent by CLP and MDA+CLP, the computational time of CLP seem able to be improved if an good initial solution having low travel distance is used. This observation matches the hypothesis given by Saab (1994) stating that the running time of LESS and FAST algorithms that CLP was developed based on could be improved if a good initial solution having low cost (i.e. referred as distance in this context)is fed into the algorithm. However, a definitive conclusion on this hypothesis is still hard to be made, since the differences are not always significant and for 3 out of 30 cases, the combined algorithm still spent longer time than CLP to solve.

Yet, the combined algorithm shows promise that a good solution could be obtained within a practical timeframe, since the computational times of 20 out of 30 cases were reduced after applying the combined method. Even though the results yielded by MDA+CLP ranged from 0.1% to 8% among all 30 cases, the improvement still prevails. Also, due to the nature of CLP that iteratively searches for better permutations (sequences), CLP has a good chance to produce better results than methods like COI, OOS and even MDA. Furthermore, by combining the advantage of MDA that can obtain good solutions quickly, there is an incentive to use the combination of these algorithms to minimize the total travel distance of item pickers.

In the next section, this concept was tested by observing the quality of the layouts generated according to the item sequences created by MDA and MDA+CLP in order to see how well the concept of linear placement can be applied to the design of item storage layouts.

5.3 Phase 2: layout generating

5.3.1 Overview

In phase two, item storage layouts are generated based on the item sequences obtained in the previous phase. The layouts are then compared with the ones generated by COI and OOS with respect to total travel distance. Since the results of MDA and CLP obtained in the previous phase were not significantly different, the item sequences arranged by MDA and the combined algorithm are the only ones used in this phase. The results are given as the percentage improvement for the layouts generated by MDA over the outcomes of COI or OOS. Also, for further analysis, detailed results are given in Appendix A providing the travel distances according to the layouts generated by MDA, MDA+CLP, COI and OOS.

A sensitivity analysis that involves solving the problems with three different rack sizes was performed in order to see how much the quality of the solutions or layouts designed according to the item sequence change over different rack sizes. The three sizes analyzed included: 10 bays per rack, 30 bays per rack and 50 bays per rack. Consequently, having three different rack sizes resulted in 90 different scenarios for each comparison that compares MDA or the combined algorithm against the other methods (i.e. either COI or OOS).

5.3.2 MDA versus COI and OOS

Each sequence was used to create three different layouts where each has a different rack size. The percentage improvements of MDA over COI and OOS are given in Tables 5.1 and 5.2 for the dataset 1 and 2, respectively. For each comparison of either

MDA versus COI or MDA versus OOS, there are 90 scenarios with different rack sizes, and number of items and order types. Moreover, the percentages are placed into the columns headed by the rack size (i.e. 10, 30 and 50) accordingly. Note that the highlighted values are set to zero because one rack is big enough to hold all items. This implies that picking any order in any layout will result in the same amount of distance traveled, since a picker is forced to traverse through the aisle once he/she start picking an order (i.e. S-shape routing policy).

In addition to the tables, the percentage differences are plotted against each other in Figures 5.13 and 5.14 for dataset 1, in order to illustrate the percentage improvement fluctuation over different rack sizes. This type of plot is also done in Figures 5.15 and 5.16 for dataset 2.

Table 5.1 Percentage Improvements of MDA over COI and OOS when the rack size is set to 10, 30, and 50 for Dataset 1

Case No.	Number of Items	Number of Order Types	MDA Percentage Improvement					
			MDA VS COI			MDA VS OOS		
			10	30	50	10	30	50
1	26	20	-0.55	0.00	0.00	0.00	0.00	0.00
2	44	30	-8.58	-0.16	0.00	-4.34	-0.16	0.00
3	91	50	6.65	0.75	0.00	7.68	1.98	0.00
4	65	60	-5.32	-0.17	-0.02	-6.99	-0.59	0.00
5	138	80	-6.92	-6.20	-3.97	-8.74	-2.79	-2.43
6	306	100	-15.51	-4.93	-0.70	-17.61	-4.17	-1.22
7	493	1173	-16.37	-18.16	-17.74	-17.12	-18.74	-17.52
8	278	1841	-4.21	1.83	0.20	-5.68	1.38	-0.03
9	509	3031	-23.80	-22.15	-23.02	-24.43	-22.87	-24.28
10	486	4164	-20.08	-25.20	-23.68	-19.93	-25.51	-24.27
11	512	5797	-25.92	-25.85	-29.40	-27.47	-27.66	-31.35
12	524	6500	-17.88	-22.33	-25.46	-16.37	-20.28	-21.95
13	528	7023	-19.02	-24.27	-23.62	-19.23	-22.19	-22.64
14	533	8769	-20.89	-26.06	-24.86	-20.92	-26.61	-25.05
15	530	9728	-18.88	-22.12	-23.23	-18.79	-21.76	-23.18
16	537	15374	-19.27	-23.55	-23.34	-19.53	-24.06	-24.05
17	538	20624	-18.03	-21.76	-22.40	-18.82	-22.30	-23.38
18	528	33373	-14.82	-23.50	-24.46	-14.29	-21.74	-22.45
19	553	42048	-19.31	-22.07	-23.38	-19.28	-22.13	-23.69
20	553	51221	-19.41	-23.36	-25.21	-19.49	-22.50	-25.36
21	557	60257	-18.27	-22.01	-23.58	-18.61	-22.37	-23.70
22	567	94079	-17.96	-21.23	-23.18	-17.45	-19.69	-23.08

Table 5.2 Percentage Improvements of MDA over COI and OOS when the rack size is set to 10, 30, and 50 for Dataset 2

Case No.	Number of Items	Number of Order Types	MDA Percentage Improvement					
			MDA VS COI			MDA VS OOS		
			10	30	50	10	30	50
1	60	75	-38.30	-1.31	-0.52	-34.36	-1.31	-0.83
2	100	125	-33.41	-27.79	-0.52	-30.35	-27.92	-0.64
3	199	239	-32.29	-35.96	-21.10	-27.96	-27.51	-18.00
4	400	421	-25.55	-41.15	-36.95	-22.71	-35.64	-33.86
5	600	680	-28.01	-44.52	-46.64	-25.45	-37.46	-40.60
6	800	841	-24.66	-41.76	-44.75	-22.27	-35.98	-37.92
7	1000	1250	-27.26	-44.65	-49.94	-24.44	-38.83	-44.23
8	1497	1610	-22.58	-37.95	-44.74	-20.46	-32.15	-37.96

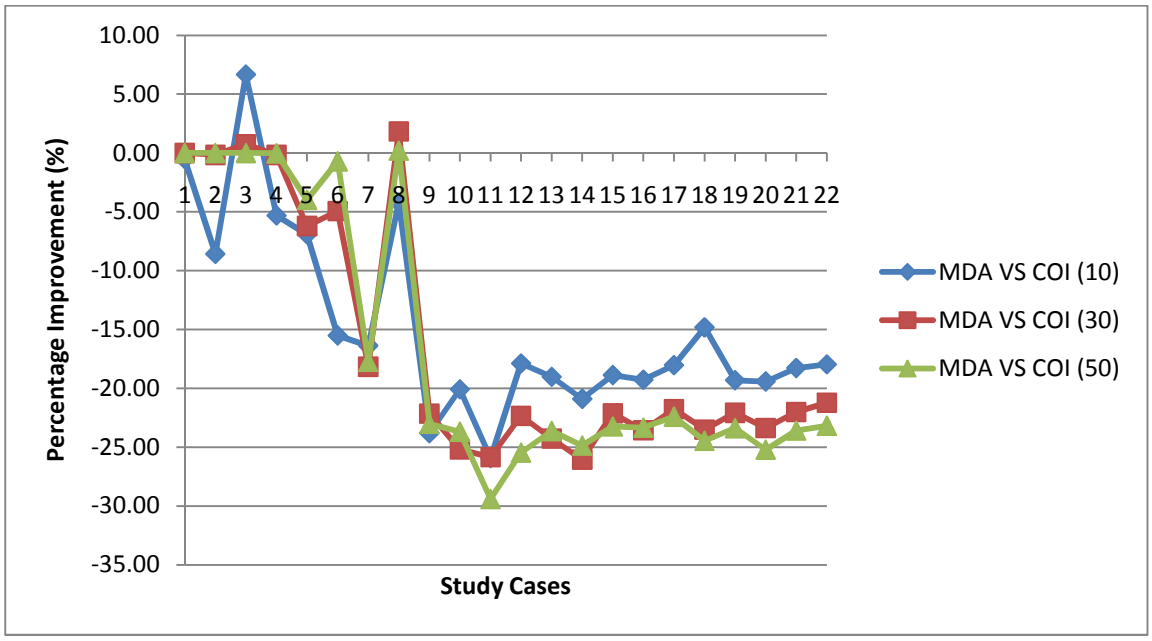

Figure 5.13 Percentage improvements of MDA over COI when the rack size is set to 10, 30 and 50 for Dataset 1

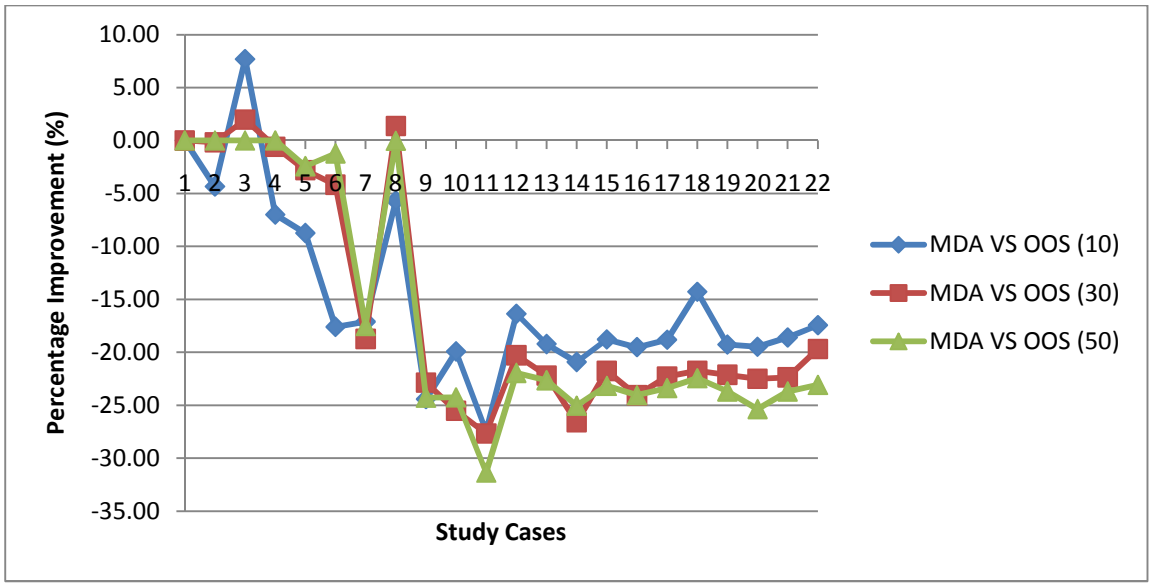

Figure 5.14 Percentage improvements of MDA over OOS when the rack size is set to 10, 30 and 50 for Dataset 1

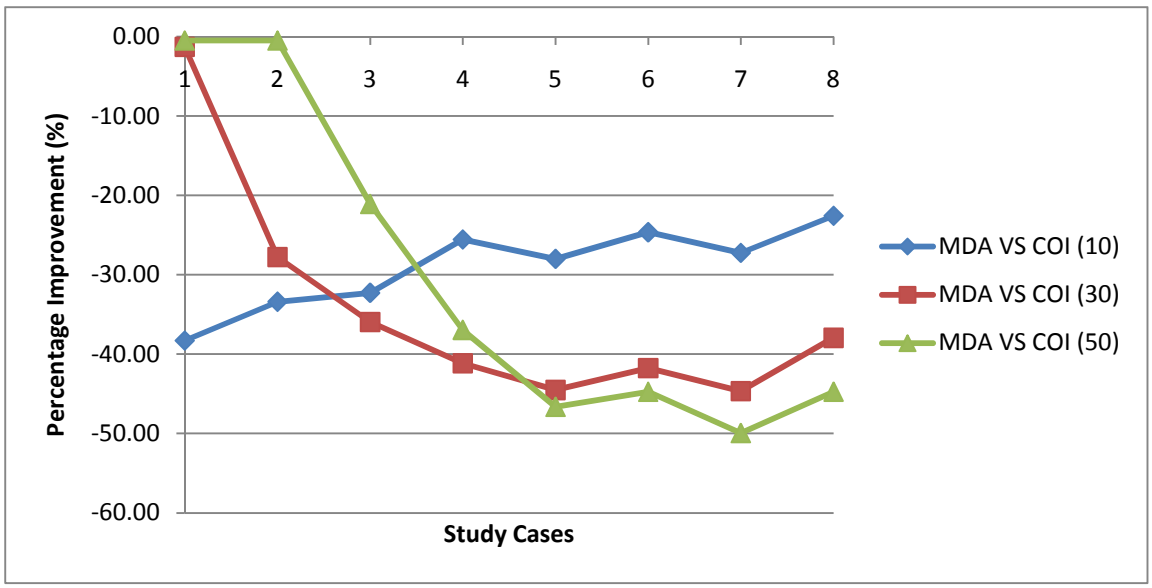

Figure 5.15 Percentage improvements of MDA over COI when the rack size is set to 10, 30 and 50 for Dataset 2

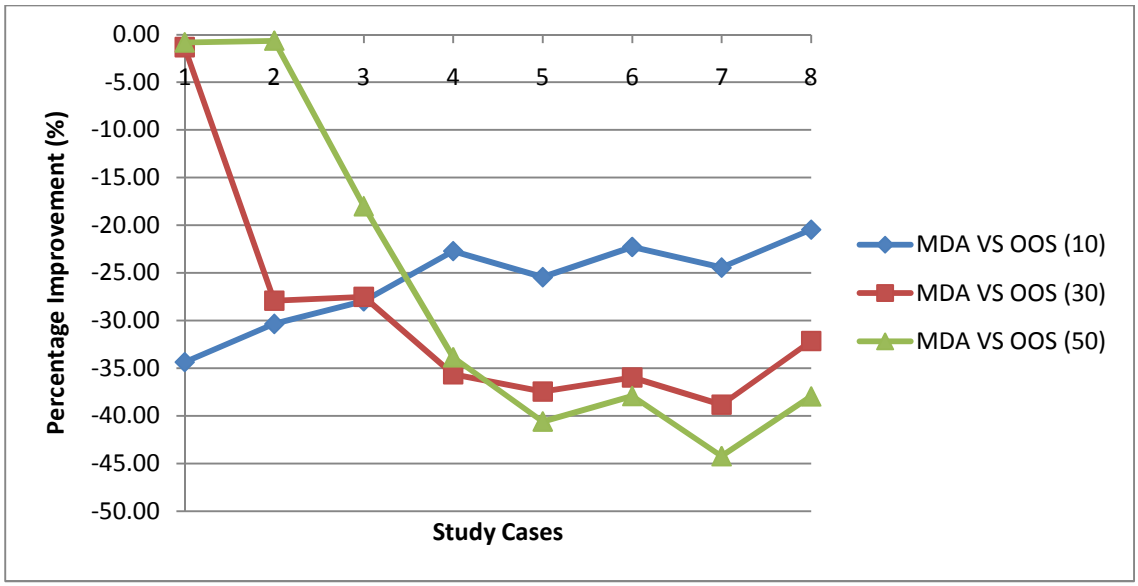

Figure 5.16 Percentage improvements of MDA over OOS when the rack size is set to 10, 30 and 50 for Dataset 2

According to the percentage differences shown above, the results among the three different types of scenarios defined by the three different rack sizes (i.e. presented as three different lines) seem to follow the same direction. Even though in dataset 2 the gaps between the lines seem wider than those in dataset 1, the lines have similar trend in the negative Y region. From this we can conclude that the concept of using an item sequence seems able to be applicable for the item storage assignment problem even when the rack size changes. Note that in the first and second cases of dataset 2, shown in Figures 5.15 and 5.16, the reason that the percentages when the rack size equals 50 are significantly far from other points for different rack sizes could be that the size of the problems are small compared to the rack size itself. For example, a rack having a size of either 30 bays or 50 bays can contain almost every items involved in that problem, so the walking distances caused by the layouts generated by MDA, OOS or COI are not much different.

Out of 90 scenarios there are four scenarios that COI produced a lower distance than MDA (i.e. the percentages are positive); however, the differences are not significantly as they ranged from 0.2% to 7.6%. For all the remaining scenarios MDA produced less distance than COI. Similarly, OOS produced three layouts having less travel distance than the ones generated by MDA, but the percentage differences are also small, compared to the other scenarios that MDA yielded up to a 44% saving.

The seven scenarios where OOS and COI produced better results than MDA involved only two out of the 30 different study cases (i.e. case number 1 and 8). These cases are relatively small compared to the other cases that may contain up to 1500 items or 94000 order types. Even though there is no proof that the size of problem affects how well COI and OOS perform, a small number of items or order types could mean that the item relationship is not well defined. Therefore, as MDA focuses on how each item relates to others and tries to first place worst items to worst locations, using COI to place the most popular items with respect to their required space would make more sense if the item relationship is not strongly exist. In addition, OOS may gain an advantage in this type of scenario that has weak relationships by considering the popularity of each item pair and then placing most popular items to best locations.

The proof of such statement is beyond the scope of this thesis. However, the results obtained in the experiments empirically show that this concept does affect the total travel distance of item pickers.

5.3.3 MDA+CLP versus COI and OOS

In this section, the results of the combined algorithm are compared against the outcomes of COI and OOS. Tables 5.3 and 5.4 show the percentage differences for datasets 1 and 2, respectively. Likewise, in Figure 5.17 to 5.20, these values are plotted to graphically present how much the results change for different rack sizes.

Again, the negative percentage represents the case where the combined algorithm yielded a better result than the other methods. Also, the highlighted numbers are the cases where one rack is big enough to contain all items, such that the distances resulting from applying each algorithm are all the same.

Table 5.3 Percentage Improvements of MDA+CLP over COI and OOS when the rack size is set to 10, 30, and 50 for Dataset 1

Case No.	Number of Items	Number of Order Types	MDA+CLP Percentage Improvement					
			MDA+CLP VS COI			MDA+CLP VS OOS		
			10	30	50	10	30	50
1	26	20	-1.14	0.00	0.00	-0.60	0.00	0.00
2	44	30	-7.87	-0.63	0.00	-3.59	-0.63	0.00
3	91	50	-0.21	0.56	0.00	0.75	1.78	0.00
4	65	60	-5.60	-0.17	-0.02	-7.27	-0.59	0.00
5	138	80	-5.62	-7.18	-3.92	-7.46	-3.81	-2.37
6	306	100	-17.46	-5.51	-1.63	-19.51	-4.76	-2.14
7	493	1173	-16.51	-15.76	-16.57	-17.26	-16.36	-16.35
8	278	1841	-4.15	-0.50	0.40	-5.62	-0.94	0.17
9	509	3031	-23.82	-24.04	-25.60	-24.46	-24.75	-26.82
10	486	4164	-21.06	-26.65	-27.32	-20.90	-26.96	-27.88
11	512	5797	-26.22	-26.33	-28.36	-27.75	-28.13	-30.34
12	524	6500	-17.80	-23.04	-23.23	-16.30	-21.00	-19.62
13	528	7023	-18.03	-20.89	-21.27	-18.24	-18.72	-20.25
14	533	8769	-21.36	-25.03	-24.59	-21.40	-25.60	-24.77
15	530	9728	-18.83	-21.90	-23.13	-18.75	-21.55	-23.07
16	537	15374	-18.91	-22.73	-22.55	-19.17	-23.24	-23.27
17	538	20624	-18.39	-21.31	-22.36	-19.17	-21.86	-23.33
18	528	33373	-17.43	-24.75	-23.61	-16.91	-23.02	-21.57
19	553	42048	-19.51	-21.53	-23.18	-19.48	-21.59	-23.50
20	553	51221	-18.69	-21.57	-23.31	-18.77	-20.69	-23.46
21	557	60257	-18.45	-21.31	-23.63	-18.79	-21.67	-23.75
22	567	94079	-17.80	-20.69	-22.91	-17.29	-19.15	-22.81

Table 5.4 Percentage Improvements of MDA+CLP over COI and OOS when the rack size is set to 10, 30, and 50 for Dataset 2

Case No.	Number of Items	Number of Order Types	MDA+CLP Percentage Improvement					
			MDA+CLP VS COI			MDA+CLP VS OOS		
			10	30	50	10	30	50
1	60	75	-41.37	-1.31	-0.52	-37.63	-1.31	-0.83
2	100	125	-35.72	-26.36	-0.76	-32.78	-26.48	-0.89
3	199	239	-35.91	-38.81	-22.67	-31.82	-30.74	-19.63
4	400	421	-31.15	-44.75	-40.31	-28.52	-39.57	-37.39
5	600	680	-31.53	-46.80	-48.04	-29.09	-40.04	-42.16
6	800	841	-28.11	-43.83	-46.87	-25.84	-38.26	-40.30
7	1000	1250	-29.92	-46.64	-51.58	-27.21	-41.03	-46.05
8	1497	1610	-26.38	-41.00	-47.05	-24.36	-35.48	-40.55

Figure 5.17 Percentage improvement of MDA+CLP over COI when the rack size is set to 10, 30 and 50 for Dataset 1

Figure 5.18 Percentage improvement of MDA+CLP over OOS when the rack size is set to 10, 30 and 50 for Dataset 1

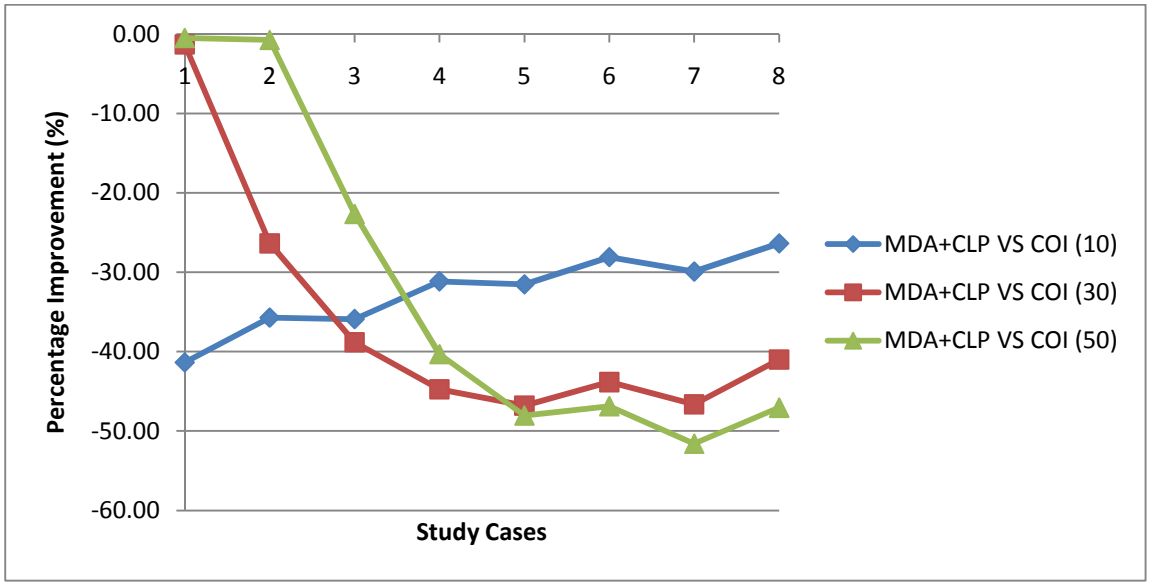

Figure 5.19 Percentage improvement of MDA+CLP over COI when the rack size is set to 10, 30 and 50 for Dataset 2

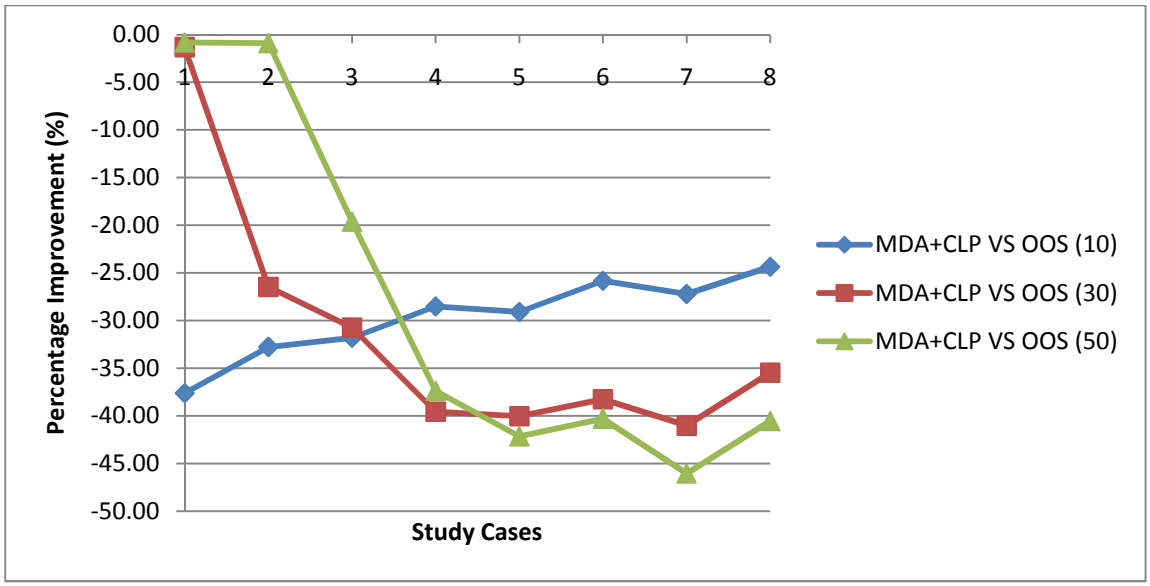

Figure 5.20 Percentage improvement of MDA+CLP over OOS when the rack size is set to 10, 30 and 50 for Dataset 2

Since the results of MDA+CLP are not much different from the ones generated by MDA alone, the charts look similar to those in section 5.3.3. According to these percentages, COI and OOS produced better results in five out of 180 scenarios. This implies that MDA+CLP improved the MDA's solutions so that the number reduced from seven scenarios, shown in the previous section, to five scenarios. Again, only two study cases are involved in these five scenarios. However, the range of these five scenarios' percentages reduced to 0.17% to 1.78%. It was found that up to a 50% and 46.8% distance reduction could be achieved by comparing the combined algorithm with either COI or OOS on the dataset 2, respectively.

5.3.4 Cross result discussion

Referring to the results obtained so far from sections 5.3.2 and 5.3.3, the travel distance has been improved from the ones generated by MDA after applying the

combined algorithm. The improvement ranged from 1% to 6% and averaged 2%. Even though MDA+CLP did not eliminate the two cases where COI and OOS yielded better results, the highest percentage difference was reduced from 7.7% to 1.8%. MDA and MDA+CLP produced better results on all other study cases. Therefore, both methods have a high chance of producing better results than COI and OOS, since more than 90% of the study cases had better outcomes by using these two methods.

According to the results from sections 5.3.2 and 5.3.3, the reduction in travel distance seems more visible when the rack size and problem size increase. As shown in the figures of both sections, when the number of items grows such that several racks need to be used, the resulting differences between MDA and the other methods are larger when there are 30 or 50 bay rack than with a 10 bay rack. The reason for this might be that the item locations dominate the amount of distance required to complete an order pick as the rack size and number of racks increase (i.e. since the number of items increases). In other words, the travel distance of an order can be minimized if all items belonging to an order are located in the same aisle. Therefore, as the rack size increases, the concept of placing related items closed to each other becomes more possible. Note that the item relationship is assumed to exist.

Finally, when the size of problems increases, the distance saving offered by MDA or MDA+CLP over COI or OOS are more pronounced, as can be seen with the lines in Figures 5.13 to 5.20 sloping down. The reason for this could be that the item relationship is more perceptible in larger size problems with a large number of order types than in the smaller problems, since there are more order types for each item to be associated with.

Chapter 6 Conclusion and Directions for Future Research

6.1 Summary and Conclusions

The objective of this research was to develop a heuristic algorithm that considers the relationship among items while minimizing the travel distance of item pickers. The research modified the concept of linear placement to arrange SKUs, so that highly correlated items are placed near each other and I/O location. Later, the placement was used to orderly assign items to real storage locations.

A method termed, Minimum Delay Algorithm (MDA), generates sequences of items effortlessly in term of computational time. Meanwhile, it provided results comparable to those yielded by CLP, which is an iterative improvement algorithm that was originally used to design VLSI circuit gate-array. In addition, to gain the advantages of both methods, MDA was used to create an initial solution that was later iteratively improved by CLP. It was found in most of the study cases that the computational times of CLP that utilized the initial solutions from MDA were lower than the computational times of CLP using randomly generated solutions. Also, in almost every study case examined in this research, the item storage layouts integrating the concept of linear placement resulted in significantly less travel distance than those generated by COI and OOS.

Even though MDA+CLP did not significantly outperform MDA alone, the nature of CLP that iteratively improves a previously obtained solution, plus the speed of MDA providing a good initial solutions in a short amount of time, creates an incentive for using the combined algorithm in future study cases or even real problems. However, the 30

study cases used in this research may not be enough to prove the real performance of this method. In contrast, both MDA and MDA+CLP show that the integration of the item relationship and the item storage assignment does provide significant savings in travel distance, which is a major cost in the picking process.

6.2 Direction for Future Research

6.2.1 Quality of solutions

As it was mentioned before, the number of study cases analyzed in this report may not be large enough to guarantee the performance of the proposed methods when they are applied to real industrial data. Therefore, more experiments on real industrial problems need to be done in order to confirm its value and whether it reflects real industrial operational decisions such as picking policy and routing policy. In addition, even though the combined algorithm was able to improve the solutions obtained by MDA, the final outcomes may or may not be significantly less than the initial solutions. In order to confirm its performance, more study cases including real industrial data is needed.

6.2.2 Integration of other warehousing aspects

Since the travel distance does not only depend on where each item is allocated, other warehousing decisions such as picking and routing policies need to be considered (Peterson et al., 2004). Therefore, the need to identify how the concept of item relationship in the item storage assignment integrates with other warehousing decisions while considering issues beyond travel distance such as aisle congestion, ease of item administration, warehouse and rack layouts, and restocking accessibility. Some examples

of the integration of warehouse operational decisions can be found in several sources (Caron et al., 2000; Peterson et al. 2004a, 2004b, 2005; Gu et al., 2007).

REFERENCES

CARON, F., MARCHET, G. & PEREGO, A. 1998. Routing policies and COI-based storage policies in picker-to-part systems. *International Journal of Production Research,* 36**,** 713 - 732.

CARON, F., MARCHET, G. & PEREGO, A. 2000. Optimal layout in low-level picker-to-part systems. *International Journal of Production Research,* 38**,** 101 - 117.

DE KOSTER, R., LE-DUC, T. & ROODBERGEN, K. J. 2007. Design and control of warehouse order picking: A literature review. *European Journal of Operational Research,* 182**,** 481-501.

GU, J., GOETSCHALCKX, M. & MCGINNIS, L. F. 2007. Research on warehouse operation: A comprehensive review. *European Journal of Operational Research,* 177**,** 1-21.

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

HWANG, H., YONG HUI, O. & CHUN NAM, C. 2003. a stock location rule for a low level picker-to-part system. *Engineering Optimization,* 35**,** 285.

JEWKES, E., LEE, C. & VICKSON, R. 2004. Product location, allocation and server home base location for an order picking line with multiple servers. *Computers & Operations Research,* 31**,** 623-636.

KALLINA, C. & LYNN, J. 1976. Application of the Cube-Per-Order Index Rule for Stock Location in a Distribution Warehouse. *INTERFACES,* 7**,** 37-46.

KANG, S. 1983. Linear ordering and application to placement. *Proceedings of the 20th Design Automation Conference.* Miami Beach, Florida, United States: IEEE Press.

MANTEL, R. J., SCHUUR, P., C. & HERAGU, S. S. 2007. Order oriented slotting: a new assignment strategy for warehouses. *European Journal of Industrial Engineering,* 1**,** 301-316.

OHTSUKI, T., MORI, H., KUH, E., KASHIWABARA, T. & FUJISAWA, T. 1979. One-dimensional logic gate assignment and interval graphs. *Circuits and Systems, IEEE Transactions on,* 26**,** 675-684.

PETERSEN, C. G. & AASE, G. 2004. A comparison of picking, storage, and routing policies in manual order picking. *International Journal of Production Economics,* 92**,** 11-19.

PETERSEN, C. G., AASE, G. R. & HEISER, D. R. 2004. Improving order-picking performance through the implementation of class-based storage. *International Journal of Physical Distribution & Logistics Management,* 34**,** 534-544.

PETERSEN, C. G. & SCHMENNER, R. W. 1999. An Evaluation of Routing and Volume-based Storage Policies in an Order Picking Operation. *Decision Sciences,* 30**,** 481-501.

PETERSEN, C. G., SIU, C. & HEISER, D. R. 2005. Improving order picking performance utilizing slotting and golden zone storage. *International Journal of Operations & Production Management,* 25**,** 997-1012.

SAAB, Y. & CHEN, C.-H. 1994. An Effective Solution to the Linear Placement Problem. *VLSI Design,* 2**,** 117-129.

SAAB, Y. & RAO, V. Year. Linear ordering by stochastic evolution. *In:* VLSI Design, 1991. Proceedings., Fourth CSI/IEEE International Symposium on, 4-8 Jan 1991 1991. 130-135.

SAAB, Y. G. 1996. An improved linear placement algorithm using node compaction. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on,* 15**,** 952-958.

TREVINO, S., WUTTHISIRISART, P., NOBLE, J. & CHANGE, A. 2008. An approach for order-based warehouse slotting. *MU CELDi Working Paper, University of Missouri, Columbia, MO*.

YAMADA, S., OKUDE, H. & KASAI, T. 1989. A hierarchical algorithm for one-dimensional gate assignment based on contraction of nets. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on,* 8**,** 622-629.

APPENDICES

Appendix A: Linear Placement Results on Dataset 1

Node	Net	MDA		CLP		MDA + CLP	
		Result	Time	Result	Time	Result	Time
26	20	102740	0	102674	0	97856	0
44	30	339650	0	309178	0	312202	0
91	50	845274	0	854764	0	822419	0
65	60	768963	0	767280	0	758660	0
138	80	1349253	0	1279508	0.2	1333371	0
157	100	1265879	0	1259726	0.2	1244182	0
493	1173	65831380	0	67400278	60	65435544	15
278	1841	116830037	0	120133936	26	115092795	7
509	3031	158805726	0	155676922	262.8	156276372	177
486	4164	403697703	0	398541638	367.4	396394141	147
512	5797	566533092	1	568428325	418	554838843	517
524	6500	420704924	2	414594392	2023.6	408895370	1165
528	7023	823272898	2	825664515	1237	813241592	823
533	8769	1042290927	2	1011457386	1226.4	1028169089	794
530	9728	1179753633	3	1211840130	1562.4	1178438035	730
537	15374	1834344490	4	1885695369	2312	1824241762	1436
538	20624	2494257383	8	2540835897	3298	2490686163	1442
528	33373	4053841138	14	3820787194	6229.2	3746567284	8171
553	42048	5335143462	24	5339931024	15037.8	5272537578	5433
553	51221	6589029584	31	6567951318	10494.6	6491539184	13956
557	60257	7825010058	35	7851926170	18207.6	7794224954	9000
565	94079	12680348382	70	13058087714	16424	12665659544	9993

Node	Net	COI		OOS	
		Result	Time	Result	Time
26	20	106772	0	99782	0
44	30	390422	0	390170	0
91	50	912056	0	912246	0
65	60	805617	0	861716	0
138	80	1637547	0	1659135	0
157	100	1692984	0	2094125	1
493	1173	84001984	0	86885430	6
278	1841	132851979	0	136709501	7
509	3031	227901204	0	241148430	36
486	4164	519342300	0	544685968	24
512	5797	838102317	0	896840684	68
524	6500	522585402	0	530763828	122
528	7023	1065949845	0	1104294387	106
533	8769	1325931305	0	1382605366	142
530	9728	1494401940	0	1542624943	146
537	15374	2366225145	0	2456714908	390
538	20624	3154979212	0	3277959116	612
528	33373	4591628520	0	4716151270	2155
553	42048	6894200566	0	7128141548	4698
553	51221	8494853050	0	8778188742	5519
557	60257	9956954408	0	10263285488	7961
565	94079	16253918642	0	16697407036	18167

Appendix B: Linear Placement Results on Dataset 2

Node	Net	MDA		CLP		MDA + CLP	
		Result	Time	Result	Time	Result	Time
60	75	2429	0	2332	0.00	2334	0
100	125	6475	0	6218	0.00	6213	0
199	239	23036	0	22713	0.00	22500	0
400	421	84635	0	80184	0.00	79063	1
600	680	193669	0	188485	1.40	187644	1
800	841	327653	0	315122	2.20	314772	1
1000	1250	608749	1	598664	5.00	588769	3
1497	1610	1162794	0	1132511	10.00	1114107	7

Node	Net	COI		OOS	
		Result	Time	Result	Time
60	75	2895	0	2954	0
100	125	7671	0	7883	0
199	239	28429	0	27728	0
400	421	97263	0	98052	0
600	680	240928	0	242027	0
800	841	396218	0	393651	0
1000	1250	761840	0	753006	1
1497	1610	1423621	0	1410894	1

Appendix C: Storage Layouts on Dataset 1

Node	Net	MDA		
		10	30	50
26	20	235104	603384	992664
44	30	485444	1124076	1821108
91	50	1136402	1768914	2767660
65	60	919822	1946008	3124368
138	80	1510354	2423376	3805902
306	100	1554708	3058080	4727276
493	1173	54564276	57120244	68036760
278	1841	135260310	152504282	171775190
509	3031	148988696	160366736	181459156
486	4164	337987214	342872158	413716682
512	5797	510496576	550813080	614010860
524	6500	355426736	360510560	400552348
528	7023	697047954	696653492	809811108
533	8769	845407728	841789494	998359812
530	9728	955202582	981943752	1138045340
537	15374	1491634394	1516677252	1771798568
538	20624	1986437290	2026989528	2353781086
528	33373	3163994988	3079700888	3540428796
553	42048	4442002948	4564923296	5203450516
553	51221	5428787004	5507091980	6241825508
557	60257	6392490792	6452496492	7408343864
565	94079	10251423312	10489724440	11909672944

Node	Net	MDA+CLP		
		10	30	50
26	20	233696	603384	992664
44	30	489220	1118804	1821108
91	50	1063290	1765482	2767660
65	60	917090	1946008	3124368
138	80	1531572	2398050	3808254
306	100	1518842	3039474	4683290
493	1173	54473384	58792840	69004068
278	1841	135350070	149018082	172119510
509	3031	148938072	156460276	175374856
486	4164	333864094	336219866	393980096
512	5797	508469932	547270604	623108336
524	6500	355761608	357252824	412492280
528	7023	705597416	727688926	834826508
533	8769	840341838	853403908	1001976758
530	9728	955715278	984621944	1139541786
537	15374	1498145696	1533006984	1789901562
538	20624	1977803814	2038512932	2355148704
528	33373	3067173436	3029213836	3580270972
553	42048	4430764084	4596605704	5216907076
553	51221	5477726388	5635431640	6400663116
557	60257	6378276388	6510343640	7403132880
565	94079	10271684076	10560619624	11951498948

Node	Net	COI		
		10	30	50
26	20	236400	603384	992664
44	30	531000	1125916	1821108
91	50	1065510	1755682	2767660
65	60	971510	1949264	3125128
138	80	1622718	2583618	3963422
306	100	1840070	3216590	4760776
493	1173	65246800	69792556	82707516
278	1841	141207644	149767250	171431704
509	3031	195513092	205987548	235734572
486	4164	422917888	458356356	542096764
512	5797	689131566	742820260	869734556
524	6500	432825228	464181008	537335696
528	7023	860763040	919866006	1060307398
533	8769	1068657064	1138400540	1328685454
530	9728	1177477294	1260787300	1482498748
537	15374	1847600542	1983887028	2311127982
538	20624	2423445114	2590697362	3033325742
528	33373	3714417012	4025733116	4686645056
553	42048	5504848432	5857976168	6791252300
553	51221	6736620624	7185567136	8345693648
557	60257	7821558032	8273448860	9694219064
565	94079	12496347976	13316259608	15503179300

Node	Net	OOS		
		10	30	50
26	20	235104	603384	992664
44	30	507444	1125900	1821108
91	50	1055334	1734522	2767660
65	60	988994	1957496	3124368
138	80	1655086	2492998	3900700
306	100	1886906	3191308	4785716
493	1173	65834300	70289472	82487392
278	1841	143403102	150433604	171824530
509	3031	197161140	207918532	239647404
486	4164	422094682	460307398	546275814
512	5797	703804938	761443486	894435944
524	6500	425024272	452232984	513186840
528	7023	862989424	895339794	1046816930
533	8769	1069093632	1147019134	1331965642
530	9728	1176262584	1255019460	1481352322
537	15374	1853552536	1997138366	2332854116
538	20624	2446944072	2608633468	3071937520
528	33373	3691512192	3935029568	4565083476
553	42048	5502640792	5862027988	6819115508
553	51221	6743184224	7105686288	8362757056
557	60257	7854288368	8311803736	9708971932
565	94079	12418314976	13061269124	15484161344

Appendix D: Storage Layouts on Dataset 2

Node	Net	MDA		
		10	30	50
60	75	2610	4810	7686
100	125	5406	8350	13014
199	239	15080	18398	26428
400	421	44630	39048	52000
600	680	95332	70834	87740
800	841	152640	99310	117646
1000	1250	273532	161784	179466
1497	1610	509282	265978	271598

Node	Net	MDA + CLP		
		10	30	50
60	75	2480	4810	7686
100	125	5218	8516	12982
199	239	14272	17578	25902
400	421	41278	36662	49226
600	680	90670	67918	85432
800	841	145642	95776	113130
1000	1250	263528	155982	173598
1497	1610	484312	252910	260252

Node	Net	COI		
		10	30	50
60	75	4230	4874	7726
100	125	8118	11564	13082
199	239	22270	28728	33494
400	421	59950	66354	82474
600	680	132426	127668	164426
800	841	202590	170526	212936
1000	1250	376028	292318	358522
1497	1610	657842	428680	491500

Node	Net	OOS		
		10	30	50
60	75	3976	4874	7750
100	125	7762	11584	13098
199	239	20932	25380	32230
400	421	57746	60670	78620
600	680	127874	113266	147704
800	841	196384	155116	189508
1000	1250	362022	264492	321790
1497	1610	640304	392004	437782