8-2016

Time Based Modeling of Storage Facility Operations

Nadeepa Devapriya Wickramage

Clemson University

Follow this and additional works at: https://tigerprints.clemson.edu/all_dissertations

Recommended Citation
https://tigerprints.clemson.edu/all_dissertations/1690

This Dissertation is brought to you for free and open access by the Dissertations at TigerPrints. It has been accepted for inclusion in All Dissertations by an authorized administrator of TigerPrints. For more information, please contact kokefe@clemson.edu.
TIME BASED MODELING OF
STORAGE FACILITY OPERATIONS

A Dissertation
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
Industrial Engineering

by
Nadeepa Devapriya Wickramage
August 2016

Accepted by:
Dr. William G. Ferrell, Jr., Committee Chair
Dr. Scott J. Mason
Dr. Kevin M. Taaffe
Dr. Amin Khademi
Abstract

This dissertation studies routing in storage facilities with stackable unit loads with time as the objective function. In the storage facilities, orders need to be stored as well as picked and material handling vehicles can carry more than one unit load at a time if the unit load containers are stackable on each other. In the first part of this research where the number of unit loads the vehicle can carry is limited to two and only one unit load can be stored at a location, a mathematical model was developed to find optimal paths for multi-command operations to route the unit load handling vehicle that minimizes total travel time of the vehicle and unit load handling time. Time savings from the proposed model can be over 25% compared to the single unit load handling single command heuristics. Three heuristics were also developed that give sub optimal solutions, yet can be used to get quicker solutions for larger problems.

Exploring further, routing in unit load storage facilities when the number of unit loads a vehicle can carry is not limited to two and when more than one unit load can be stored at a location is studied. A mixed integer linear programming model is developed and four route construction heuristics are presented to construct routes that minimizes the total time. The heuristic that constructs routes based on operating time time and starting from the locations with longest operating time provides best routes.

Routing methods in temporary storage facilities where unit loads can be shipped to alternate destinations and have limited storage capacity and operation time is studied. A construction heuristic is proposed to make internal routing, storing, and reshipping decisions.
Dedication

Dedicated to everyone who helped.
Acknowledgments

My PhD journey would not have been possible if not for the sacrifices, support and guidance of many people who actively contributed to the successful completion of my thesis. I owe my heartfelt gratitude to those who were by my side, patiently and resiliently, throughout this journey.

First of all, I would like to thank my research advisor Professor William Ferrell for his continued guidance provided throughout this journey. This thesis would not have materialized if not for his meticulous contributions in shaping and refining the ideas that originated through the discussions we had. His support, dedication and patience made my journey bearable. I would also like to show my gratitude to Dr. Scott Mason, Dr. Kevin Taaffe, Dr. Amin Khademi and Dr. Maria Mayorga for their invaluable support and guidance that they offered to me whenever I reached out to them for help.

I would like to thank the Department of Industrial Engineering for providing me this opportunity for my graduate studies. My sincere gratitude goes to all the faculty and administrative staff of the Department of Industrial Engineering for providing productive and supportive environment. None of this would have been possible without the wonderful teachers I had at the University of Moratuwa and the Royal College, Colombo who inspired me to pursue knowledge.

I’m thankful to all my friends for your unwavering friendship which enabled me to endure all the hard times. Finally, I cannot thank enough my loving wife Dulma, my son and daughter, my parents, and family for standing by me all the time and making it possible for me to pursue this PhD.
Table of Contents

Title Page ................................................................. i
Abstract ................................................................. ii
Dedication ............................................................... iii
Acknowledgments ......................................................... iv
List of Tables ............................................................. vii
List of Figures ............................................................ viii

1 Introduction .......................................................... 1
   1.1 Unitload Storage Facility ......................................... 2
   1.2 Research Motivation ............................................... 6
   1.3 Research Contributions .......................................... 7
   1.4 Dissertation Organization ........................................ 8

2 Literature Review ..................................................... 9
   2.1 Storage ............................................................ 9
   2.2 Order Picking .................................................... 11
   2.3 Multiple Unit Load Handling ..................................... 17
   2.4 Intermodal Facility Operations ................................... 19

3 Time-Based Multi-Command Operations in Unit Load Warehouses . 23
   3.1 Introduction ....................................................... 23
   3.2 Background ....................................................... 25
   3.3 Time-Based Optimization Model .................................. 27
   3.4 Numerical Study ................................................... 35
   3.5 Constrained Optimization ........................................ 50
   3.6 Construction Heuristics ......................................... 54
   3.7 Conclusions and Future Work ..................................... 60

4 Multi-Command Routing Operations in Unit Load Storage Facilities 62
   4.1 Introduction ....................................................... 62
   4.2 Background ....................................................... 63
   4.3 Multi-Command Optimization Model .............................. 65
   4.4 Heuristic Methods ................................................ 70
## List of Tables

<table>
<thead>
<tr>
<th>Number</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Total route time for 20 location problems</td>
<td>37</td>
</tr>
<tr>
<td>3.2</td>
<td>Comparisons of heuristic results for the 9 pallet problem</td>
<td>52</td>
</tr>
<tr>
<td>3.3</td>
<td>Comparison of total time and computing time for constrained optimization methods</td>
<td>52</td>
</tr>
<tr>
<td>3.4</td>
<td>Comparison of total time and computing time for heuristic methods</td>
<td>57</td>
</tr>
<tr>
<td>3.5</td>
<td>Comparison of total time and computing time for best methods</td>
<td>58</td>
</tr>
<tr>
<td>3.6</td>
<td>Comparison of location visiting order in heuristic methods</td>
<td>59</td>
</tr>
<tr>
<td>4.1</td>
<td>Travel and handling times for 20 location problems</td>
<td>81</td>
</tr>
<tr>
<td>4.2</td>
<td>Total route time for 20 location problems</td>
<td>84</td>
</tr>
<tr>
<td>4.3</td>
<td>Comparison of heuristic results with exact method results</td>
<td>86</td>
</tr>
<tr>
<td>4.4</td>
<td>Comparison of total route time with model characteristics</td>
<td>87</td>
</tr>
<tr>
<td>4.5</td>
<td>Parameters of pick and store demand lists</td>
<td>107</td>
</tr>
<tr>
<td>4.6</td>
<td>Average computation times each heuristic</td>
<td>110</td>
</tr>
<tr>
<td>5.1</td>
<td>Total reshipped unit loads for limited outbound capacity</td>
<td>140</td>
</tr>
<tr>
<td>1</td>
<td>Distance matrix for a 20 location problem</td>
<td>174</td>
</tr>
<tr>
<td>2</td>
<td>Stackability matrix for a 20 pallet problem</td>
<td>175</td>
</tr>
<tr>
<td>3</td>
<td>Sample set of routes generated by the MILP for a 20 pallet problem</td>
<td>175</td>
</tr>
<tr>
<td>4</td>
<td>Distance matrix and store &amp; pick list for a sample 20 location problem</td>
<td>176</td>
</tr>
<tr>
<td>5</td>
<td>Sample routes generated by <em>Shortest-time location visit</em> algorithm</td>
<td>176</td>
</tr>
<tr>
<td>6</td>
<td>Total route time vs. unit load pick time</td>
<td>177</td>
</tr>
<tr>
<td>7</td>
<td>Total route time vs. unit load store time variation</td>
<td>177</td>
</tr>
<tr>
<td>8</td>
<td>Total route time vs. unit load store and pick time</td>
<td>177</td>
</tr>
<tr>
<td>9</td>
<td>Total route time vs. vehicle speed</td>
<td>178</td>
</tr>
<tr>
<td>10</td>
<td>Total route time vs. vehicle speed - minimum &amp; maximum values</td>
<td>178</td>
</tr>
<tr>
<td>11</td>
<td>Total route time vs. vehicle capacity</td>
<td>178</td>
</tr>
<tr>
<td>12</td>
<td>Total route time vs. vehicle capacity - minimum &amp; maximum values</td>
<td>179</td>
</tr>
<tr>
<td>13</td>
<td>Total route time vs. number of locations</td>
<td>179</td>
</tr>
<tr>
<td>14</td>
<td>Total route time vs. number of locations - minimum &amp; maximum values</td>
<td>179</td>
</tr>
<tr>
<td>15</td>
<td>Total route time vs. input range</td>
<td>180</td>
</tr>
<tr>
<td>16</td>
<td>Sample inbound &amp; outbound schedule in an intermodal facility</td>
<td>181</td>
</tr>
<tr>
<td>17</td>
<td>Sample reship cost matrix of an intermodal facility</td>
<td>182</td>
</tr>
</tbody>
</table>
## List of Figures

1.1 Example of reusable pallet size containers ................................................. 3  
1.2 Stacked unit load transportation ............................................................... 4

3.1 Feasible activity-paths ............................................................................ 29  
3.2 Layout of 9 pallet example ...................................................................... 36  
3.3 Pallet stackable density variation .............................................................. 39  
3.4 Vehicle speed variation ........................................................................... 41  
3.5 Assymetric vehicle speed variation ............................................................ 43  
3.6 Pallet handling time variation .................................................................. 45  
3.7 Warehouse size variation .......................................................................... 47  
3.8 Warehouse aisle layouts ........................................................................... 48  
3.9 Warehouse aisle layout variation ............................................................... 49  
3.10 Average number of multi-command operations and solve time vs. number of pallets ................................................................. 50

4.1 Objective function value variation with time in the MILP method .................. 80  
4.2 Total route time for 20 location problems ................................................. 84  
4.3 Total route time statistics for solution methods ......................................... 88  
4.4 Total route time vs. unit load pick time variation in 20 location problems . 89  
4.5 Travel time vs. unit load pick time ............................................................ 90  
4.6 Total route time vs. unit load store time .................................................... 92  
4.7 Total route time vs. unit load pick and store time ..................................... 95  
4.8 Total route time vs. vehicle speed ............................................................. 97  
4.9 Minimum and maximum route times vs. vehicle speed ............................. 99  
4.10 Total route time vs. vehicle capacity ....................................................... 100  
4.11 Minimum and maximum route times vs. vehicle capacity ....................... 102  
4.12 Total time vs. number of locations .......................................................... 104  
4.13 Minimum and maximum route times vs. number of locations ................. 106  
4.14 Total route time vs. store and pick demand ............................................. 108

5.1 A single-stage intermodal interface layout ................................................. 125  
5.2 Number of vehicles variation with unlimited outbound and unlimited storage capacity ................................................................. 133  
5.3 Minimum and maximum time, number of trips, and cost vs. number of vehicles ................................................................. 135  
5.4 Maximum storage vs. number of vehicles for limited outbound capacity .... 137  
5.5 Maximum number of stored units in each time period ............................ 138  
5.6 Total cost vs. number of vehicles for limited outbound capacity ............... 139
Chapter 1

Introduction

Manufacturers and markets are more distributed in current global economy and global logistics continues to grow. Logistics accounts for a significant fraction of the gross national product. In the U.S.A. transportation sector represents 10% of the U.S. Gross Domestic Product, which is approximately $1.6 trillions in 2012 according to Department of Transportation [16]. While transportation and packaging takes majority of the expenditure on transportation sector, a significant amount is spend on commercial warehousing.

Warehouses, in general storage facilities, are an important element in logistics systems. As of 2003, there were nearly 600,000 warehouses in United States according to a Energy Information Administration report (2006) [7]. Efficient storage facility operations is essential in facilitating the providing fast and reliable delivery of goods. Order picking operations in storage facilities such as warehouses accounts for no less than 55% [26] of the total operational costs. Therefore, order picking is one of the most critical processes in a warehouse. Maximizing the service level is the most common objective of order-picking systems [48]. The faster an order can be picked, the sooner it can be shipped to the customers and thus increases service level. Therefore, minimizing the order picking time is a requirement for order picking systems. Since travel time is 50% of the order picking time [117], most of the studies on warehouse design and optimization consider travel
distance as the primary objective despite various studies have shown activities other than travel may substantially contribute to order picking time [41], [38]. Minimizing the total time considering travel time and goods handling time can be more direct and effective way to improve service level.

1.1 Unitload Storage Facility

Goods handled at commercial storage facilities can range from small packages (less than unit load size) up to fully loaded containers, also known as pallets or unit loads. Small packages handling has been improved with advances in automated storage retrieval systems (AS/RS) in parts-to-picker systems. Yet, unitload containers remain as an important component of picker-to-parts storage facilities. Developing efficient methods to handle fully loaded unit loads is essential for efficient supply chains. This research focuses on material handling in unit load warehouses.

Material handling equipment such as forklifts are used to handle unit load containers. Most material handling equipment is operated by humans and are non-automated. Automated Guided Vehicles (AGV) are also used to transport pallets. Common AGV applications have repetitive movements and regular delivery of stable loads. However, this research focuses on non-automated, human operated vehicles because most movements are less likely to be repetitive. For efficient operation of storage facilities, routing directions can be given to human operators of material handling equipment similar to automated counterparts.

As a part of sustainable goods movement, containers with lesser impact on environment and land use are encouraged. Eco-friendly, reusable containers with a footprint of a standard pallet which have been designed to be safely stacked on each other are already being used in the industry. Figure 1.1 [1] shows an example of reusable pallet size containers. Ability to vertically stack unit load containers opens up an opportunity to reduce container handling time by transporting multiple containers in a single trip.
1.1.1 Single, dual and multi-command operations

Among many warehouse functions such as unloading, checking, stock-storing, packing, etc. Storing and order-picking are two major operations. In common warehouse terminology storing or picking one item on a trip is called single command operation. When storing and picking operations are combined on a trip, it is referred as dual command operation. In unit load warehouses where a material handling vehicle can carry only one unit load per trip all trips are either single command or dual command trips.

However, if more than one unit load can be carried in a trip more opportunities to store or pick unit loads in a single trip arise. In that case, several locations can be visited to store or pick unit loads in a single trip depending on the capacity of the material handling vehicle. When more than one pallet is stored or picked in a single trip, the operation can be referred as multi-command operations. As intuition suggests, multi-command operations can combine several dual command or single command trips and as a result, travel time and unit load handling time may be saved. As a result multi-command operations can improve efficiency of warehouse operations. In this research, we
explore the use of multi-command operations in various storage facilities. Figure 1.2 [2] shows a vehicle carrying two stacked unit loads in a warehouse with more than one unit load stored at a location.

Figure 1.2: Stacked unit load transportation

1.1.2 Storage space and unit load handling

Warehouse storage space is a limited resource and therefore using the space efficiently to store unit loads is a common goal for warehouse management. The simplest way of storing unit loads is on the floor, typically arranged in lanes. The storage locations can be accessed via access aisles. However, storing unit loads directly on the floor is not efficient in terms of storage space. To increase the available storage space efficiently unit loads may be stored on each other or on supporting racks. When multiple unit loads may be stored in a single location, locations and unit loads do not have a one-to-one mapping.

When determining storage and order-pick routes, storage capacity at a location can be a decisive factor. If more than one unit load can be stored at a location, a location can have both store and pick demand. A single trip might be able to fulfill the store and pick demand of a location given that the vehicle can carry multiple unit loads and unit
loads are stackable. If the number of unit loads that can be stored at a location exceeds the vehicle capacity, more than one trip might be required to fulfill the store and pick demand. On the other hand, handling multiple unit loads in a single trip would reduce the number of trips and eventually reducing the travel time. Furthermore, handling multiple unit loads would save time compared to handling them individually. These factors need to be taken into consideration when designing routes for store and order-pick operations.

In this research, we study storage facilities with both characteristics: single unit load per location, also known as one-to-one mapping between store-pick list and locations, and multiple unit loads per location.

1.1.3 Storage facilities with temporary storage

Storage and pickup operations are not limited to commercial warehouses. Facilities such as cross-docking facilities and less-than-truckload breakbulk terminals store goods temporarily, generally less than a day. Goods typically leave a cross-dock in less than one hour within arrival. These temporary storage facilities, too, use storage and pickup operations extensively to route goods from the inbound dock doors to the outbound dock doors. Intermodal transportation, in which two or more modes of transportation are used for delivering goods from origin to destination, has evolved into hub based network structure. Small sized shipments such as unit loads or pallets are consolidated at intermodal hubs and shipped between hubs using intermodal containers.

A potential next-generation logistics system that features extensive collaboration, generally known as Physical Internet uses next-generation intermodal interfaces. In such facilities, small sized shipments may reach the final destination routing through shared logistics facilities without pre-determined routes. Therefore it is important to understand the characteristics and develop operational methods for intermodal facilities where re-shipping goods to alternate destinations is possible. Limited storage quantity and time restrictions in such facilities make goods handling challenging.
1.2 Research Motivation

Improving performance of storage facilities is enforced by increasing cost reduction targets due to market competition and supply chain integration. Determining efficient methods to route material handling vehicles to pick items stored in a storage facility, known as the “order picking problem” has been a key focus area in storage facilities related research. These facilities usually operate with deadlines, hence they are driven by time constraints.

However, majority studies on order picking problem use distance minimization approaches to find routes and do not consider time for handling the goods, which might be a significant portion of the total time. Furthermore, use of reusable and stackable containers opens up an opportunity to transport more than one unit load in one trip which enables multi-command operations. Multi-command operations lead to handling multiple unit loads at the store or pick location, which can save handling time compared to handle unit loads individually. Therefore, it is important to consider time as an objective in determining routes for storing and order picking. The literature on time related studies in routing is limited to vehicle routing problems and its variations. However, research on time minimizing routing in storage facility related operations where routes are between a central depot and storage or pick locations has a gap in the literature.

Furthermore, temporary storage facilities that operates with time limitations such as cross-docks and intermodal interfaces heavily use material handling vehicles and handle unit loads. These facilities, too, operate with time limitations. The literature on temporary storage facilities address strategic, tactical and operational problems. Operational problems focus on problems such as pick up and delivery vehicle routing, inbound and outbound truck scheduling with and without time windows and temporary storage problems. Most of the studies assume fixed internal goods handling time and ignore the possibility to reship goods and storage limitations. Therefore, it is important to study the operations in facilities with limited storage and possibility of reshipping.
1.3 Research Contributions

Many models found in the literature use graph theory based distance minimization approach to solve order picking problem which does not consider unit load handling time, nor allows to study characteristics of unit loads such as stackability. In this research, operations in a unit load operation facilities with stackable unit loads is considered. In the first part of the research, finding optimal routes that minimize total travel and unit load handling time with multi-command operations in one-to-one location to unit load mapped warehouse setting where material handling vehicle can carry two or lesser number of unit loads will be tested with the proposed optimization model. Real-world limitations for determining routes such as loaded vehicle’s speed variations and unit load stackability variations will be considered in the proposed optimization model.

Subsequently, warehouses that do not have one-to-one location to unit load mapping, i.e. warehouses with more than one unit load stored at a location, is studied where material handling vehicles can carry more than two unit loads at a time. Although there are similar exact and heuristic approaches in the vehicle routing problem domain, none has considered time components for handling goods at the locations and visiting a location multiple times to fulfill the store and pick demand of a location. In this research, exact method for this problem shall be presented and heuristic approaches shall be studied to find solutions for large problem instances.

In temporary storage facilities such as intermodal interfaces, limited storage capacity and time sensitivity are key differences in contrast to traditional warehouse operations. Internal operations to consolidate shipments when departure deadlines are involved and when the goods can be reshipped to an alternate destination when storage capacity is limited is a special problem category associated with next-generation logistics systems. In this research, a practical model for internal operations of an intermodal interface is be presented.
1.4 Dissertation Organization

The remainder of this dissertation is organized as follows. Chapter 1 introduces the dissertation and provides a background and the motivation for this work, including an introduction to the research question and the structure of this dissertation. Chapter 2 discusses the background knowledge of the work in the fields of warehouse store and pick operations, pick and delivery in vehicle routing problem, and cross-dock operations. While Chapter 2 explores the broad research methodologies used in this work, each chapter elaborates on specific research techniques used in the context of that work.

Chapter 3 presents the time-based models for multi-command operations in storage facilities where only one unit load can be stored at a location and vehicles carry two unit loads at a time. A mixed integer linear programming model and route construction heuristic methods are presented for the above problem. Numerical study is performed to validate the model and compare the heuristic methods.

Chapter 4 extends the time-based modeling to generalized storage facilities where more than one unit load can be stored at a location and vehicles may carry more than two unit loads at a time. A mixed integer linear programming model is presented and four route constructing heuristics are introduced along with numerical study and results.

Chapter 5 proposes a heuristic solution method to make routing, storing, and shipping decisions in an intermodal hub where temporary storage capacity is limited and unit loads need to be transported from inbound truck containers to outbound truck containers within a given time frame while reshipping to alternate destinations is allowed. A numerical study is conducted to validate the model and study the correlations of parameters on decisions.

Chapter 6 summarizes the conclusions derived from each of the studies performed in this research. A list of recommended future research on time based modeling of storage facilities is also provided in this chapter.
Chapter 2

Literature Review

This work is focuses on three types of storage systems: Warehouse storage systems with single unit load stored at a location, warehouse storage systems with multiple unit loads stored at a location, and temporary storage facilities with reshipping capability. Following sections of this chapter provides an overview of the literature to each of the storage systems.

2.1 Storage

Literature on commercial storage systems discusses many problems and present different models to support solve them. Literature reviews on warehouses systems has been performed. For example, Berg and Zijm (1999) [122] classifies warehouse management problems and discusses order picking models. Gu et al. (2007) [53] conducted extensive review on warehouse operations. De Koster et al. (2007) [39] gives an general overview on decision problems in manual order picking process at warehouses. Gu et al. (2010) gives an overview on warehouse design and performance evaluation [54]. Warehouse design and planning problems can be classified in to two basic problems; warehouse design and warehouse operation. In this research, the focus will be on warehouse operation and in particular, storage and order picking.
Order receipt is the first step of warehouse inbound operation. When goods are received to the warehouses, they need to be stored before shipped. In literature, storage assignment and layout planning problems are well documented. In this research, storage assignment or layout planning problems will not be focused. However, following section summarizes key research areas about warehouse storage.

Most studies found in literature have focused on handling stock keeping units (SKUs) individually, which is different from unit-load operations. The major decision making criteria in storage assignment and layout problems can be identified as storage efficiency and access efficiency [53].

How to assign SKUs to storage locations is studied in storage assignment strategies. Since storage location of SKUs affect the order picking operation, the storage decisions may be based on order picking efficiency. The simplest form of storing is randomly store inbound SKUs to available storage locations. Despite random storage is simple, this method may result in longer expected travel time and distance (Choe and Sharp, 1991)[70]. However, random storage can be be used with a system to track the inventory accurately using computer control [39]. Other popular strategies are class-based and dedicated storage.

Gu et al. [53] classifies storage location problem according to information available in the problem. When complete information about items, that is arrival and departure time of each item is known, problem becomes an assignment problem. However, more than one item can use the same storage location, but not at the same time. When only product information, that is characteristics such as product size, usage rate, are known, problem becomes assignment of items to product classes and assignment of product classes in to storage locations and known as Class-Based Storage.

Class-based storage is one of the storage strategies used in practice. Hausman et al. (1976)[60] suggested class-based storage to reduce the expected picking travel time by locating high-demand products near input/output point point (depot) in an automated warehouse. A strategy to reduce total pick time is to divide the warehouse into a forward
area and reserve area. Hackman and Rosenblatt (1990) [57] develop a greedy heuristic model to decide which SKUs to be assigned to forward area. The objective is to maximize the total benefit of the forward area. van den Berg et al. (1998) [124] propose a model for a unit-load warehouse with busy and idle periods where units in forward area can be replenished instantaneously. Idle time prior to the busy period is used to replenish in advance so that number of replenishments during busy period can be reduced. Gu et al. (2009) [55] use a branch-and-bound algorithm to solve the forward-reverse problem optimally. In above models, storage within a class or area is random. A special case of class-based storage is when number of classes is equal to number of products. The policy is called Dedicate Storage.

In Dedicated storage strategy, different methods can be used to assign an item to storage locations. Popular rule is cube-per-order index (COI), which is the ratio of the space requirement (cube) of a SKU to its turnover rate (popularity). Heskett (1963) [67] first introduced the COI policy. Implementation of COI policy is discussed in Kalinna and Lynn (1976) [71]. COI policy is proven optimal in minimizing material handling cost in dedicated storage under certain conditions. For example, Malmborg and Krishnakumar (1989) [80] prove COI policy is optimal for multi-command order picking warehouses when certain assumptions satisfied. For dedicated storage, several heuristic algorithms can be found in literature in addition to COI policy. For example, Zhang et al. (2002) [130] uses genetic algorithm based heuristic for storage location assignment problem in multiple-level warehouses. There are various heuristic approaches proposed in literature, but are not discussed here.

### 2.2 Order Picking

In most warehouses, order picking requires significant amount of capital investment or significant amount of labor (Goetschalckx and Ashayeri, 1989)[48]. Order picking accounts for no less than 55% (Bartholdi and Hackman, 2010) [26] of the total operational
warehouse costs. Therefore, order picking is one of the most critical processes in a warehouse. Many papers in literature discusses order picking cost minimizing methods.

Orders may be picked according to customer orders by pallets, by cases (cartons) or by individual SKUs. Many different order picking systems are used in warehouses. When the package size is small, usually manual pickup or Automated Storage/ Retrieval System (AS/RS) is used. AS/RS systems are used when material handling volume is large. To handle fully loaded pallets, material handling equipment such as fork lifts are used. Single warehouse may employ more than one order-picking system. There are numerous papers that discuss parts-to-picker systems which includes AS/RS and manual picker-to-parts systems.

One common objective of order picking systems is to maximize service level subject to constraint of resources such as labor, capital and machines (Goetschalckx and Ashayeri, 1989) [48]. Service level is calculated combining different factors such as order delivery time, order accuracy etc. Minimizing order picking time is one of the methods to ensure higher service level. Order picking time reduction through reducing travel distance is achieved using four methods in the literature namely (1) optimizing order picking routes, (2) warehouse zoning, (3) batching orders, and (4) storage-location assignment. The focus of this research would be optimizing order picking routes. Two approaches to optimize order picking routes can be found in literature. One approach is to minimize travel distance while other approach is to minimize total time including travel time and handling time.

2.2.1 Distance based models for order picking

During a typical order pick process, 50% of the time is used to travel between part location and depot according to Tompkins et al. (2003) [117]. A case study by Dekkrr et al. (2004) [41] support this claim. Although nearly half of the time for a pick process is spent for tasks other than travelling, many models in the literatures uses models to minimize travel distance.
When travel distance increases, travel time increases for warehouses with manual order picking. Therefore, travel distance is selected as the primary objective in optimization. Determining the best sequence and route to pick or storing locations can be considered as a Traveling Salesman Problem (TSP) which is specific to warehouse operations. The problem is specific to warehouses because of the layout of the warehouse, travel paths are limited. Ratliff and Rosenthal (1983) [106] propose a polynomial-time dynamic programming exact algorithm to solve this problem in a warehouse with parallel multi-aisles. The rectangular shaped warehouse has one depot and the aisles are connected by a cross aisle at each end and is considered a single block warehouse. The objective is to minimize distance travelled by forklift when given items are picked and transported the items to shipping area. The solution from their model is optimal with computational time linear in the number of aisles.

Goetschalckx and Ratliff (1988) [49] used a graph theory based approach for routing order pickers in a warehouse with wide-single cross aisle. The pickers pick less than pallet load items and return to the vehicle in this model.

Optimal algorithms are not available for every warehouse layout. Heuristic algorithms provide good routes for order pickers. For example, Petersen (1997) [101] carried out number of numerical experiments to compare routing methods and compared the heuristic solution with optimal solution. Petersen and Asse (2004) [102] used simulation to examine how the picking, storage and routing decisions effect order picker travel.

Daniels et al. (1998) [37] used a TSP based heuristics with local search feature to model warehouse order picking. The model determines the sequence of visiting given picking locations. In 2010, Theys et al. [116] examine LinKernighanHelsgaun (LKH) TSP heuristic, developed by Helsgaun (2000) [62] to solve the sequencing and routing of order-pickers in a warehouse.

Order picking problem has been associated with other warehouse problems. For instance, Gray et al. (1992)[51] develop a model to solve warehouse layout, equipment and technology selection, item location, zoning, picker routing, pick list generation and
order batching problems. Tsai et al. (2008) [118] address the order picking problem in relation to the order batching problem and developed an algorithm that searches for the most effective travel path for a batch by minimizing the travel distance. Despite these models give a good insight to the order picker travel problems, this research will look at the total travel time and handling of the order picker.

2.2.2 Time based models for order picking

Literature on time based models mostly focus on AS/RS where vehicle travels on a single aisle than manual material handling warehouses. For example Bozer and White (1984) [32] developed travel time models for unit load automated AS/RS performing both single and dual command cycles for randomized storage conditions. Sarker and Babu (1995) [111] reviews the design aspects of AS/RS travel time models.

In the area of non-automated order picking systems, Hall (1993) [58] compares different routing heuristics. The author uses the route-length estimates to compare these routing policies across a variety of warehouse configurations. De Koster and van der Poort (1998) [40] extended algorithm developed by Ratliff and Rosenthal (1983) [106] for a warehouse where order pickers can start and return to the head of any aisle instead of to the depot. De Koster and van der Poort (1998) [40] used graph theory as solution approach and constructed a dynamic programming algorithm for calculating order picking tours of minimal length in warehouses. Further, Roodbergen and de Koster (2001) [107] extended the algorithm by Ratliff and Rosenthal (1983) [106] for a warehouse with three cross aisles that divides the warehouse to two blocks. In addition, in 2001, Roodbergen and De Koster [108] used graph theory as time based solution approach and extended the work by De Koster and Van der Poort (1998) [40]. In this model, order pickers can change aisles at the end of every aisle and also at a cross aisle halfway along the aisles. Roodbergen and Vis (2006) [109] developed travel-time models for warehouses considering two routing policies. Travel-time estimates of a picking route using a non-linear optimization model is used by the authors to determine the layout of a order picking area consisting of one
Queueing theory is another approach used to solve order picking problems. For instance, Chew and Tang (1999) [34] modelled order picking systems in a single block warehouse as a queueing system and used simulation method to compare and validate results. Work by Chew and Tang (1999) [34] was extended by Le-Duc and de Koster (2007) [75] by considering a 2-block warehouse and performing a direct analysis on the average throughput time of a random order when less than full pallets are picked. Yu and De Koster (2009) [129] proposed a G/G/m queueing network approximation model to analyze the impact of order batching and picking area zoning on the mean throughput time in an a pick-and-pass order picking system. Parikh and Meller (2010) [97] developed analytical expressions to estimate throughput based on probability models and order statistics results assuming random storage. Malmborg and Al-Tassan (2000) [81] studied the impact of item, equipment, storage configuration and operating parameters using an integrated model in less than unit load order picking systems. They combined the travel time and storage space models to estimate order picking cycle times time models for single and dual command operations.

All the papers listed above do not consider the possibility of stacking pallets. Hassan and Ferrell (2010) [18] developed a model to include stackability to minimize total travel distance that allowed multiple picks in any route. They extended stackability model to two separate time-based models for pallet picking and pallet storage. The objective of the time based models is to minimize total time including travel time and pallet handling time.

### 2.2.3 Combined Storage and Picking

Literature on combined storage and picking, which pairs storage operation with a retrieval operation and also known as interleaving problem, mostly focuses on operations in automated storage and retrieval systems (AS/RS). Graves et al. (1977) [50] showed that total travel distance can be reduced by planning storage and picking together to reduce
unproductive travel between storage and retrieval locations. In this research, interleaving would be used for multi-command problems.

In order to minimize travel distance between storage location and retrieval location, Han et al. (1987) [59] suggested to match storage location and retrieval location. Lee and Schaefer (1997) [76] presented an sequencing method for static assignment problem in a unit load AS/RS with dedicated storage, which can be solved in polynomial time optimally. van den Berg and Gademann (1999) [123] showed that special case of sequencing under the dedicated storage policy can be solved in polynomial time. An expression for expected travel distance in a warehouse that uses dual-command operations was developed by Pohl et al. (2009) [103] and analyzed three common warehouse layout designs. Hassan and Ferrell (2010) [18] modeled a unit load warehouse with stackable pallets to find routes for multi-command operations by minimizing total travel and handling time.

### 2.2.4 Stacked Unit Load Handling

Safety is a highly concerned aspect in warehouse operations. In the US, nearly 35,000 workplace related injuries have reported in warehouse and storage sector, in a single year (U.S. Department of Labor, 2010) [13]. Another report shows that more than 1000 workers have died from injuries in forklift-related incidents from 1980 to 1994 (CDC, 2000) [6]. Overturns is a major cause for forklift accidents and excessive speed is one of the reasons for overturns. (CDC, 2000) [6]. Larsson and Rechnitzer (1994) [74] suggested to control speed of forklifts when they are closer to pedestrian workers. Horberry et al. (2004) [68] propose to limit forklift speed especially when they are loaded in order to enhance forklift safety. In some cases, speed of forklifts may be mechanically limited when they are carrying a heavier load [68]. It is advised to drive the forklift in reverse if the load blocks the field of vision of the forklift driver, for instance when carrying stacked full pallets [12] [10] [8].

When travelling reverse, travelling speed is reduced compared to forward travel. Moreover, when carrying two stacked pallets, reducing the travelling speed is advised to
prevent overturns and tipping. There are no time-based route optimization models in the literature that incorporates different speeds for carrying stacked and unstacked pallets. In this research, this gap will be fulfilled by extending the model by Hassan and Ferrell (2010) [18] to study how does reverse travel when pallets are stacked affect routing decisions.

2.3 Multiple Unit Load Handling

Vehicle Routing Problem (VRP) is a classical optimization problem. In VRP, the goal is to assure pickup and/or delivery of goods in a distribution network where cyclical routes starting and ending in a depot are determined. A generalized problem of VRP is Pickup and delivery problem (PDP). In PDP goods are picked up at one location and delivered to another location. Pickup and delivery problem has been studied. Savelsbergh and Sol (1995) [112] present a survey of the problem types and solution methods for Pickup and Delivery Problem. Parragh et al. (2007) [98] and (2008) [99] presents a comprehensive survey on pickup and delivery models. First part of the survey [98] discusses problems with the transportation of goods from the depot to linehaul customers and from backhaul customers to the depot, also known as Vehicle Routing Problems with Backhauls (VRPB). Four subtypes of VRPB are identified in the survey. The second part of the survey [99] discusses problems where goods are transported between pickup and delivery locations.

Berbeglia et al. (2007) [28] surveys the methods used for solving pickup and delivery problem and introduces a tree-field classification scheme for the problem. The first field in the classification scheme is numbers of origins and destinations for the transport of commodities, for example many-to-many or one-to-one. The second field describes the pick up and delivery operations in the nodes and the third field describes the number of vehicles.

Another relaxed version of the generic VRP is split delivery vehicle routing problem (SDVRP). In this problem a delivery to a demand point can be split between any number of vehicles. The SDVRP was first introduced by Dror and Trudeau (1989) [44]. Using
a heuristic algorithm, they were able to reduce cost with split deliveries. In addition, several studies have shown the benefits of split deliveries for the VRP (Dror et al. (1994) [43], Frizzell and Giffin (1995) [47], Archetti et al. (2006) [22]). Archetti and Speranza (2012) [21] present a survey on the SDVRP.

In Split delivery vehicle routing problem, also known as VRP with backhauls, the cargo flow is unidirectional. However in Vehicle routing problem with pickup and deliveries (VRPPD) cargo flows is bidirectional where a customer is visited either for delivery, or for pickup, or both. Desaulniers et al. (2002) [42] summarizes the methods used in Vehicle routing problem with pickup and delivery (VRPPD) and examine route construction and improvement heuristics including metaheuristics and neural network approaches.

While split loads have been applied mostly to the VRP, several studies used split loads to VRP with pickup and delivery. The first paper on the VRP with split deliveries and pick-ups appeared in Mitra (2005) [86]. In the problem studied in the paper, items collected from a customer with a pick-up demand can be delivered to any customer with a delivery demand and a vehicle leaves and returns to the depot only once. Customers may be visited by more than one vehicle and more than once by the same vehicle. The problem is a many-to-many pick-up and delivery problem. Mitra presented a MILP formulation and a heuristic to solve the problem. Mitra (2007) [87] studied the same problem and proposed an alternative formulation with a parallel clustering technique.

Nowak et al. (2008) [91] introduces the pickup and delivery problem with split loads (PDPSL) which is an one-to-one pickup and delivery problem where a vehicle picks up a load from a specific origin and delivers it to its destination. They used a Tabu-search algorithm to solve the problem for large scale problem instances. It is shown that the most benefit for a set of given origins and destinations can be achieved with load sizes just above one half of vehicle capacity. Nowak et al. (2009) [92], developed an empirical analysis that using the previous heuristic. The goal of the paper was to evaluate how the benefit deriving from split loads is influenced by the mean load size and variance, by the number of origins and by the geographical distribution of origins and destinations. Thangiah et
al. (2007) [115] proposed a heuristic algorithm for SDVRP with pick-up and delivery with time windows and applied on both static and real-time data sets. The problem in this paper is a single commodity one-to-one pick-up and delivery problem similar to Nowak (2008). Tang et al. (2009) [114] used greedy algorithms and a competitive decision algorithm to solve VRP with pickup and delivery problem. Oncan et al. (2011) [96] used a branch-and-cut algorithm for Multi-vehicle One-to-one Pickup and Delivery Problem with Split Loads. Pickup and Delivery problem with Split loads has been applied to maritime applications in Andersson et al. (2011) [19]. Pickup and Delivery problem with Split loads (PDPSL) with precedence constraints (PC-PDPSL) is studied by Nowak et al. (2012) [93]. They developed a dynamic programming formulation of the PC-PDPSL and show that the state and action spaces of this problem are finite. Multi-vehicle One-to-one Pickup and Delivery Problem with Split Loads (MPDPSL) was studied by Sahin et al. (2013) [110]. In this generalized PDPSL problem, each load can be served by multiple vehicles as well as multiple stops by the same vehicle. Heuristics based on Tabu search and Simulated Annealing is proposed to solve the problem.

2.4 Intermodal Facility Operations

Transportation of goods from an origin location to a destination using at least two modes of transportation is known as intermodal transportation. The transfer from one mode to the next is performed at intermodal interfaces [35]. In global supply chains, standardized containers are used to transfer shipments between different modes of transportation such as ship, rail or truck at transfer points. In addition to the global shipping, domestic intermodal usage has been increased in recent years [15].

Intermodal hub networks consists of small number of hubs for a given region where smaller sized shipments (packages, unit loads, pallets) are consolidated at intermodal hubs. A unit load may contain number of smaller shipments each of which has a destination and possibly a delivery deadline. Each unit load also will have a destination and a
delivery dead line. Consolidated unit loads or pallets are then shipped between hubs using intermodal containers, generally known as truck-loads (TL) in road transportation operations. The transportation containers may contain a number of unit loads, each of which has a destination and possibly a delivery deadline. At a intermodal interface, many decisions must be made when a transportation container arrives at the facility to send each unit load to their destination within the delivery deadline.

Macharis and Bontekoning (2004) [79] review the application of operations research models and methods in the field of intermodal transportation. They identify four types of intermodal operators: drayage, terminal, network and intermodal operators. Drayage operators are responsible for planning and scheduling transportation between the terminal and shippers and receivers. Terminal operators arrange transhipment operations between different modes. Network operators plan infrastructure and organize long-haul container transportation. Intermodal operators use the intermodal infrastructure, services, and select routes for a shipment through intermodal network. Each operator has to make their own strategic, tactical and operational decisions.

Operational level literature on intermodal transportation includes a decision support system to assist users in selecting the least cost combination of transportation modes between an origin and a destination [29] where time and cost are minimized, and minimum cost intermodal routing on rail/road combination [24].

2.4.1 Cross-dock Operations

Although not identical, literature on cross-dock operations provide an insight on the problems faced in an intermodal interface. Belle et al. (2012) [120] provides an extensive review of the existing literature about cross-docking. Augustina et al. (2010) [17] also reviews the studies on cross-dock operations. Some of the important topics addressed in the cross-docking literature includes the strategic level decisions such as location of cross-dock facility, and layout of the cross-dock, tactical level decisions such as cross-docking networks, and operational level decisions such as vehicle routing when a cross-dock is
included, dock-door assignment, truck scheduling, and managing the temporary storage in cross-docks.

The problem where to locate cross-docks has been studied in the literature. Gümiş and Bookbinder (2004) [56] formulate optimization models to minimize total cost in multi-echelon networks with multiple origins and multiple product types. A fixed facility cost, transport cost and in-transit inventory cost is considered as the total cost. Bachlaus et al. (2008) [23] study multi-echelon supply chain with an objective of optimizing the material flow throughout the supply chain and to identify optimal number and locations of suppliers, plants, distribution centers and cross-docks. A multi-objective optimization model is used to solve the problem.

Layout of most cross-docks are I-shaped although there are various other cross-dock shapes such as L,U,T,H,E or X. Bartholdi and Gue (2004) [25] investigate the best shape for a cross-dock for performance. Based on the total internal travel distance, experiments suggest that for smaller cross-docks with fewer than 150 doors, I-shape is the most efficient; For intermediate size cross docks which have less than 200 doors, T-shape proves to be most efficient and for cross-docks with more than 200 doors, X-shape is best. Vis and Roodbergen (2008) [125] study the location of the temporary storage area for incoming unit loads such that the travel distances of the forklifts are minimized. The problem is modeled as a minimum cost flow problem.

Three basic functions can be identified in a cross-dock facility: receiving, storing, and shipping. When goods arrive at the cross-dock facility, they are scanned and verified at the receiving docks; Then the products are sorted by destination and finally sent to the outbound docks. To increase the productivity of a cross dock facility, one or more of the above operations need to be improved. Improvements in goods receiving and storing is addressed by dock door assignment and truck scheduling problems.
2.4.2 Less-than-Truckload Operations

Similar to cross-dock problems, Less-than-Truckload (LTL) problems in the literature include strategic planning problems such as design of the network, tactical planning problems such as designing the service network and determining the routes and service schedules operational planning problems such as deciding to accept or reject service requests, handling the demand and allocating resources. Crainic and Laporte [36] review the optimization based operation research methodologies for LTL operations. Some research questions related to intermodal interfaces with are addressed in LTL operations such as carrier collaboration. Data and information that are required for decision-making move backward from the lowest level (operational) to the strategic level.

Small to medium-sized LTL carriers use some forms of collaboration to address problems such as excess capacity, over lapping lanes, and facilities. By collaborating, smaller and medium-sized LTL operators increase use of capacity and facility space.
Chapter 3

Time-Based Multi-Command
Operations in Unit Load Warehouses

3.1 Introduction

This research focuses on operations in a unit load warehouse (i.e., one that only handles material on full pallets) where some or all of the unit loads can be safely stacked for movement and storage. Some of these are collapsible containers that have the footprint of a pallet but have been designed to be safely stacked while others are ordinary pallets that can be stacked simply because they contain objects that form a flat upper surface like sheet metal or sturdy boxes. In this study, all will be referred to as stackable pallets. Unit loads, interchangeably referred as pallets are moved inside the warehouse by material handling vehicles such as forklifts.

In operating warehouses today, forklift drivers frequently stack pallets so two are moved in the same trip but they select the pallets to stack based on intuition and physical proximity. Since picking operations have historically been substantially improved through the use of mathematical models, this research takes a similar approach to the problem of stackable pallets. A storage facility that uses fixed storage so each pallet has a known location is considered for this study. It is assumed that forklift operators can store and
pick pallets in the same trip and that no more than two pallets can be stacked while they are transported at one time. A trip, for example, can begin with the forklift driver leaving the depot with two pallets stacked on each other that are to be stored in known locations. The depot is a staging area in the warehouse that is sometimes called the pickup and deposit point. The first move is to store one of the pallets but the second move can either be a store or a pick. This continues until the driver returns to the depot with zero, one, or two pallets that have been picked.

The inclusion of stackable pallets demands a different approach to the routing problem from the traditional distance minimization because the fundamental nature of the problem is changed. Efficiency of the picking and put away process is evaluated based on time. Routes that require a shorter time to accomplish the task for the same number of vehicles or people equal greater efficiency. When pallets are put away and picked without stacking, minimizing the total distance traveled is equivalent to minimizing the total time required for the operations. When pallets can be stacked, this is not the case because there is time associated with manipulating stacked pallets that is not required with single high movements. For example consider the manipulations that must be performed if the lower pallet on the fork must be stored first. The driver must first set both pallets on the floor, then lift the top pallet and put it on the floor, then lift and store the lower pallet, and finally lift the remaining pallet before proceeding to the next pick or store location. Contrast this to the single high movement that does not require any of these intermediate handling steps.

Because efficiency is defined in terms of minimum time and when pallets can be stacked minimizing distance is not equivalent to minimizing time, we contend that time-based models are required. In general, the savings for any one trip might not be huge; however, as the warehouse gets larger the saving increase because stacking pallets reduces the number of trips to the depot and that time is magnified in a large warehouse. Also, high volume warehouses will also see more dramatic cost reductions simply because of the scale leverage. Regardless, this problem is theoretically interesting because the situation
demands a different modeling approach and practically interesting because the results can be a significant source of cost reduction.

Finally, the literature uses the term “dual-command” when describing storing and picking operations that are combined in one trip; however, the current understanding of dual command appears to be restricted to a trip consisting of one store followed by one pick. As such, we will use the term multi-command for trips that can have multiple picks and stores in a single trip.

### 3.2 Background

The fundamental problem underlying this research is the Order Picking Problem (OPP) that determines the route for a material handling vehicle to pick items stored in a warehouse and transport them back to the depot in an optimal way. Ratliff and Rosenthal (1983) [106] used graph theory as the solution approach to address this since their objective was to minimize distance. Since then, many interesting problems have been addressed by researchers for the myriad of practical implementations of picking found in practice as well as the many dimensions of the problem like aisle configuration and width. This has produced an incredible volume of published research so we focus attention on that of most relevance to this research.

In particular, papers that directly or indirectly determine the picking routes are of high interest. Many papers address various perturbations of the original problem using a distance-based objectives and graph theory. As argued earlier, we submit that this is not the correct objective when pallets can be stacked; rather, a time-based model must be used. Models based on travel time are not very common but a few exist. The earliest time-based models that we could find were focused on automated storage/retrieval systems (ASRS). Bozer and White (1984) [32] developed travel time models for ASRS while Sarker and Babu (1995) [111] provided a review of this segment of the literature. There are also a few models that address more general problems. Hwang and Song
(1993) [69] developed expected travel time models based on the probabilistic analysis for single and dual commands assuming randomized storage assignment policy. Queirolo et al. (2002) [105] developed a simulation model to address assigning storage areas in a warehouse to reduce travel time during the picking operation. The model reduces the global storage cost by minimizing the total travel time. Dual command operations were the subject of Pohl et al. (2009) [103] where an expression for expected travel distance was developed. They used the expression for expected travel distance to analyze three common warehouse designs and concluded that a warehouse design layout that has racks parallel to the shipping dock with aisles perpendicular to the shipping dock was best. Malmborg and Al-Tassan (2000) [81] developed an integrated model to study the impact of item, equipment, storage configuration and operating parameters in less than unit load order picking systems. They combined the travel time and storage space models to estimate order picking cycle times from which the impact of alternative interleaving disciplines can be evaluated.

Another research area that is similar to this work is the straddle carrier routing problem in a container terminal at a port. This literature focuses on minimizing the total travel time of the straddle carrier without considering the handling time of the containers. For example, Kim and Kim (1999) [72] proposed a routing algorithm for a single straddle carrier to load export containers onto a container ship.

A well-known problem in the literature is pickup and delivery problem. This problem studies the transportation of goods from the depot to line-haul customers and from backhaul customers to the depot or transportation of goods between pickup and delivery locations. Parragh et al. (2008) [99] surveyed the literature on pickup and delivery problems. This research problem has similarities to the Vehicle Routing Problem with Mixed Linehauls and Backhauls (VRPMB) in the literature where goods are transported from the depot to linehaul customers and from backhaul customers to the depot; and any sequence of linehauls and backhauls are permitted. There are several exact and heuristic approaches to the VRPMB problem. Süral and Bookbinder (2003) [113] develop a
mixed-integer model to solve a VRPMB problem. The objective is to minimize the total cost of a single tour that begins from the depot, make all deliveries, and optionally pick up some pick up demand. Several linear programming relaxations are considered and medium sized problems are solved. Nagy and Salhi (2005) [90] propose integrated construction improvement heuristic for VRPMB where backhaul customers are also considered while determining routes in contrast to insertion based methods in previous work. Pallet handling time is an important component when more than one package is handled at a location and it has not been addressed in VRPMB studies. Furthermore, VRPMB assumes that any package can be grouped with another in a route but, in warehouses, any pallet might not be able to stack on another due to stackability restrictions.

Our conclusion is that the aforementioned literature provides interesting ideas that have framed the overall approach to this research but none of the literature address the model required to adequately address pick and put away operations in a unit load warehouse operation with stackable pallets. It is this precise class of problems that the models and results presented here address.

### 3.3 Time-Based Optimization Model

This research focuses on determining the optimal picking and storage routes in a non-automated unit load warehouse that uses fixed storage. Key to this research is the fact that some or all of the pallets can be stacked on each other during movement so the time to manipulate pallets at destinations can be a significant part of total route time. Including stackability in a warehouse model has two important research consequences. First, since pallets can be stacked, it is important to include both storing and picking of multiple pallets on the same route. For example, a forklift can leave the depot with two stacked pallets to be stored. The first operation is a put but the second can be a put or pick. Clearly there are several combinations of operations that can occur before the forklift returns to the depot with zero, one, or two picked pallets. The second is that
the basic movements associated with stackable pallets induce the need for a fundamental change in the modeling approach to find an optimal route from a distance-based model to a time-based model. Specifically, it is proposed that if the overall objective is minimizing cost, then time is the critical factor. As such, the optimal solution is one that minimizes the total time and not the total distance traveled; these two objectives are not the same for unit load warehouses with stackable pallets.

A simple example of the difference can be seen by considering two pallets that must be picked that are located near the depot and next to each other. Minimizing distance would produce a route that picks the first pallet, stacks the second on top, and return to the depot. The minimum time solution time is to two single picks because the time required for stacking - sit the first pallet on the floor, back up the forklift, lift the second pallet, carefully place it on top of the first, back up the forklift, pick up both pallets and move to the depot takes longer than two single picks. And this does not include the time at the depot to pick off the top pallet and put it on the floor! In general, we think the objective has always been to minimum time and, for single pick operations, distance is a surrogate for time because the time manipulating pallets is negligible; however, when stackable pallets require minimum time approach. As such, this research proposes a time-based mixed integer programming model to determine the minimum time routes to pick and store a set of pallets in an order. The model explicitly accounts for all time elements such as time to handle two stacked pallets at the storage location, time to store, time to pick, and travel time. It is assumed that a maximum of two pallets can be stacked at one time, that storage locations can accommodate exactly one pallet, and that all pallets are available in the order are available.

The number of pallets that can be stacked on each other clearly has an impact on the effectiveness of this methodology so a new measure is defined, stackability density. This is simply the percentage of pallets that are allowed to be stacked on each other. If all pallets can be stacked on all other pallets as in a warehouse with reusable containers, the stackability is 100%. On the other hand, warehouses with traditional wooden pallets
are likely to have a much smaller fraction of stacking opportunities so the density will be much less than 100%.

### 3.3.1 Problem definition

A mixed integer programming model is proposed based on the observation that when at most two pallets can be stacked, there are only 12 possible activity-paths a forklift can take on a route. Let $S_i$ denote storing the $i^{th}$ pallet and $P_j$ be picking $j^{th}$ pallet. Then all the activity-paths for a maximum of two stacked pallets are enumerated in Figure 3.1.

#### Leave depot with TWO pallets to be stored:
- $S_1 - S_2$
- $S_1 - S_2 - P_1$
- $S_1 - P_1 - S_2$
- $S_1 - S_2 - P_1 - P_2$
- $S_1 - P_1 - S_2 - P_2$

#### Leave depot with ONE pallet to be stored:
- $S_1$
- $S_1 - P_1$
- $P_1 - S_1$
- $S_1 - P_1 - P_2$
- $P_1 - S_1 - P_2$

#### Leave depot with ZERO pallets to be stored:
- $P_1$
- $P_1 - P_2$

Figure 3.1: Feasible activity-paths

In the Figure 3.1, it is assumed that all activity-paths start and end at the depot. So, for example, $S_1-S_2-P_1$ should be interpreted as: 1) Leave depot with two pallets 1 and 2 stacked. 2) Move to the storage location of first pallet that is assume to be on the top. Lower both pallets to the floor. Pick the first pallet from the top and store it. Pick up the second pallet. 3) Move to the storage location of second pallet and store it. 4) Move to location of first pick pallet and pick it, 5) Return to the depot.
3.3.2 Assumptions

The model utilizes the following assumptions:

- All trips start and end at the depot or staging area.
- No more than 2 pallets can be stacked at one time.
- One forklift is performing the order picking and storage functions.
- Fixed storage is used meaning there is a one-to-one mapping between locations and pallets and each pallet has a unique location.
- All pallets to be stored are available at depot and those to be picked are in the storage areas throughout the warehouse.
- Stacking can be performed both at the depot and in the storage areas.
- When two pallets are stacked to be stored, the pallet to be stored first is on the top.

3.3.3 Notation

Since we are assuming fixed storage, subscripts on the variables identify pallets and their locations in the warehouse where they are to be stored or picked. There are \( n_s \) pallets to be stored which are numbered 2 through \( n_s + 1 \) and they are identified by \( i \) and \( j \) (i.e., \( i \in [2, n_s + 1], j \in [2, n_s + 1] \)). There are \( n_p \) pallets to be picked which are numbered \( n_s + 2 \) through \( n_s + n_p + 1 \) and these are represented by the indices \( k \) and \( l \) (i.e., \( k \in [n_s + 2, n_s + n_p + 1], l \in [n_s + 2, n_s + n_p + 1] \)).

The input parameters for the problem are:

- \( d_{ij} \) = the distance between location \( i \) and location \( j \) in feet
- \( S_{ij} \) = 1 if pallet \( i \) is stackable on pallet \( j \); 0 otherwise
- \( s_f \) = average speed of the forklift when carrying 0 or 1 pallet, in feet per minute
- \( s_b \) = average speed of the forklift when carrying 2 pallets, in feet per minute
- \( t_s \) = time to store a single pallet on a rack in minutes
- \( t_{ss} \) = time to store one pallet from a double stack set-up on a rack in minutes
- \( t_p \) = time to pick a pallet in minutes
- \( t_{ps} \) = time to pick a pallet and stack it on another pallet in minutes

The decision variables are all Boolean and enumerate the possible activities and routes.

- \( X_1^{ij} = 1 \) if route is store \( i \), then store \( j \); 0 otherwise
- \( X_2^{ijk} = 1 \) if route is store \( i \), then store \( j \), then pick \( k \); 0 otherwise
- \( X_3^{ikj} = 1 \) if route is store \( i \), then pick \( k \), then store \( j \); 0 otherwise
- \( X_4^{ijkl} = 1 \) if route is store \( i \), then store \( j \), then pick \( k \) then pick \( l \); 0 otherwise
- \( X_5^i = 1 \) if route is store \( i \); 0 otherwise
- \( X_6^{ik} = 1 \) if route is store \( i \), then pick \( k \); 0 otherwise
- \( X_7^{ki} = 1 \) if route is pick \( k \), then store \( i \); 0 otherwise
- \( X_8^{ikl} = 1 \) if route is store \( i \), then pick \( k \), then pick \( l \); 0 otherwise
- \( X_9^{kl} = 1 \) if route is pick \( k \), then pick \( l \); 0 otherwise

To simplify the formulation, total operation time for each decision variable are combined into model constants. For example, individual operations and time for each operation for \( X_5^{ikj} \) are as follows:

1. Pick first store pallet, time = \( t_p \)
2. Pick second store pallet and double stack, time = \( t_{ps} \)
3. Move from depot to \( i \), time = \( \frac{d_{si}}{s_b} \)
4. Store first pallet on rack, time = \( t_{ss} \)
5. Move from \( i \) to \( k \), time = \( \frac{d_{ik}}{s_f} \)
6. Pick first pick pallet and double stack, time = \( t_{ps} \)
7. Move from \( k \) to \( j \), time = \( \frac{d_{kj}}{s_b} \)
8. Store second store pallet on rack, time = \( t_{ss} \)
9. Move from \( j \) to \( l \), time = \( \frac{d_{jl}}{s_f} \)
10. Pick second pick pallet and double stack, time = \( t_{ps} \)

11. Move from \( l \) to depot, time = \( d_{l1}/s_b \)

12. Store first pic pallet from double stack at depot, time = \( t_{ss} \)

13. Store second pick pallet, time = \( t_s \)

Total time would be: \( C^5_{ikjl} = (d_{ij} + d_{kj} + d_{l1})/s_b + (d_{ik} + d_{jl})/s_f + t_p + 3t_{ps} + 3t_{ss} + t_s \)

- \( C^1_{ij} = (d_{ij})/s_b + (d_{ij} + d_{j1})/s_f + t_p + t_{ps} + t_{ss} + t_s \)
- \( C^2_{ijk} = (d_{i1})/s_b + (d_{ij} + d_{jk} + d_{j1})/s_f + 2t_p + t_{ps} + t_{ss} + 2t_s \)
- \( C^3_{ikj} = (d_{i1} + d_{k1})/s_b + (d_{ik} + d_{j1})/s_f + t_p + 2t_{ps} + 2t_{ss} + t_s \)
- \( C^4_{ijkl} = (d_{i1} + d_{j1})/s_b + (d_{ij} + d_{j1} + d_{kl})/s_f + 2t_p + 2t_{ps} + 2t_{ss} + 2t_s \)
- \( C^5_{i,jl} = (d_{i1} + d_{k1} + d_{l1})/s_b + (d_{ik} + d_{jl})/s_f + t_p + 3t_{ps} + 3t_{ss} + t_s \)
- \( C^6_{i} = (d_{i1} + d_{i1})/s_f + t_p + t_s \)
- \( C^7_{ik} = (d_{i1} + d_{ik} + d_{k1})/s_f + 2t_p + 2t_s \)
- \( C^8_{ki} = (d_{ki})/s_b + (d_{1k} + d_{i1})/s_f + t_p + t_{ps} + t_{ss} + t_s \)
- \( C^9_{ikl} = (d_{i1})/s_b + (d_{i1} + d_{ik} + d_{kl})/s_f + 2t_p + t_{ps} + t_{ss} + 2t_s \)
- \( C^{10}_{k} = (d_{k1} + d_{i1})/s_b + (d_{1k} + d_{i1})/s_f + t_p + 2t_{ps} + 2t_{ss} + t_s \)
- \( C^{11}_{k} = (d_{1k} + d_{kl})/s_f + t_p + t_s \)
- \( C^{12}_{kl} = (d_{l1})/s_b + (d_{ik} + d_{kl})/s_f + t_p + t_{ps} + t_{ss} + t_s \)


### 3.3.4 Mathematical Model

A model to minimize the total time can now be constructed. Note that in all cases, 
\( i \in [2, n_s + 1], j \in [2, n_s + 1], k \in [n_s + 2, n_s + n_p + 1], l \in [n_s + 2, n_s + n_p + 1] \)

Minimize 
\[
Z = \sum_{i,j} C_{1 ij} X_{1 ij} + \sum_{i,j,k} C_{2 ijk} X_{2 ijk} + \sum_{i,k,j} C_{3 ikj} X_{3 ikj} + \sum_{i,j,k,l} C_{4 ijkl} X_{4 ijkl} \\
+ \sum_{i,k,j,l} C_{5 ikjl} X_{5 ikjl} + \sum_{i,k} C_{6 ik} X_{6 ik} + \sum_{k,j} C_{7 kj} X_{7 kj} + \sum_{k,i,l} C_{8 kil} X_{8 kil} + \sum_{k} C_{9 k} X_{9 k} + \sum_{k,l} C_{10 kl} X_{10 kl} 
\]

subject to:
\[
\begin{align*}
X_{1 ij} & \leq S_{ij} \\
X_{2 ijk} & \leq S_{ij} \\
X_{2 ijk} & \leq S_{jk} \\
X_{3 ikj} & \leq S_{ik} \\
X_{3 ikj} & \leq S_{jk} \\
X_{4 ijkl} & \leq S_{ij} \\
X_{4 ijkl} & \leq S_{kl} \\
X_{5 ikjl} & \leq S_{ij} \\
X_{5 ikjl} & \leq S_{jk} \\
X_{5 ikjl} & \leq S_{kl} \\
X_{8 ki} & \leq S_{ki} \\
X_{9 kl} & \leq S_{kl} \\
X_{10 kl} & \leq S_{kl} \\
X_{12 kl} & \leq S_{kl} 
\end{align*}
\]
The objective function contains segments of feasible routes and the model then constructs the optimal arrangement of the segments. Constraints (3.2) eliminates routes that are not permissible because of stacking restrictions. Constraints (3.3) and (3.4) ensure each pallet is moved. Impossible moves are prohibited by (3.5). Constraints (3.6) define the nature of the decision variables.
3.4 Numerical Study

Several numerical examples are now presented to explore some of the models features. In all of these, ILOG CPLEX Optimization Studio Version 12.3 was used on a personal computer with an Intel Core i5 2.5 GHz processor, 8 GB of RAM, and running Windows 7. The forklift travel rate (speed) in a warehouse is 350 feet per minute \( (s_f = s_b = 350 \text{ ft/min}) \) ([14]). The time required to pick a pallet from the rack is 0.3 minute \( (t_p = 0.3 \text{ min}) \) ([102]). We further assume the time required to store a pallet on the rack is 0.3 minute \( (t_s = 0.3 \text{ min}) \), the time to pick a pallet and stack it on another pallet is 0.5 minute \( (t_{ps} = 0.5 \text{ min}) \) and the time to store a pallet on rack from a double stack is 0.5 minute \( (t_{ss} = 0.5 \text{ min}) \) ([18]).

This problem can be reduced to a Multiple Vehicle Routing Problem \( (mVRP) \) in which the objective is to find \( m \) tours that each start and end at the single depot node, such that each remaining node is visited exactly once and the total cost of visiting all nodes is minimized where the set of nodes represent the set of locations when each pallet is stackable on any other pallet, when vehicle capacity is 2, and when either store or pick demand is zero. VRP is a well known NP-hard problem and thus this problem is also NP-hard, solving realistic-sized problems is practically impossible and therefore the examples here are sufficiently small to find exact solutions.

3.4.1 9 Pallets in close proximity

In this example, 9 pallets are arranged as illustrated in Figure 3.2. It is assumed that \( S_{1,3} = S_{1,7} = S_{2,5} = S_{3,1} = S_{3,8} = S_{4,6} = S_{5,2} = S_{6,4} = S_{7,9} = 1 \) and all other stacking options are prohibited so the stackability density is \( 9/72 = 12.5\% \). Pallets 1 through 4 are to be stored which pallets 5 through 9 are located in the warehouse and must be picked. The distance matrix was generated based on layout where location is 5 feet long.

If these pallets are stored and picked using single unit operations, the total time is 18.8 minutes with 13.4 minutes of travel and 5.4 minutes to pick and store the pallets.
The optimal solution with stacking is 11.85 minutes with the following trips:

- Leave depot with pallet 1 - store pallet 1 - pick pallet 5 - return to depot
- Leave depot with pallet 2 - store pallet 2 - pick pallet 6 - return to depot
- Leave depot with pallet 4 - store pallet 4 - pick pallet 8 - return to depot
- Leave depot with pallet 3 - store pallet 3 - pick pallet 7 - pick pallet 9 - return to depot

The total time of 11.85 minutes includes travel time of 6.05 minutes and pallet handling time of 5.8 minutes. This simple example illustrates why optimally taking advantage of the stackable pallets can have a notable impact on efficiency since even though on 1/8th of total possibilities for stacking were available, the reduction in total travel and material handling time were reduced by about 1 minute or almost 8% when compared with single pallet operations. This provides some insight into the magnitude of the savings that can be achieved by the use of an optimal stacking strategy.

This example was constructed to test one aspect of the model and illustrate an important practical idea. Notice that the solution only stacks pallets 7 and 9 on the last trip; it does not, for example, stack pallets 1 and 3 to store these pallets. The reason is that the material handling time associated with stacking take longer than returning to the depot for the picks and stores that are located near the depot. The practical implication is that it is not better to always stack pallets. Note also that a distance based model
would have ignored the considerable handling time which, in this case, is over 50% of the
total time. We submit that this reinforces the need for development of time-based models
rather than distance based because total time is the important measure of efficiency for
a warehouse and material handling time can be significant.

### 3.4.2 20 Pallets problems

Consider a 20 pallet problem where 10 pallets are to be stored in the warehouse
and 10 pallets are to be picked from the warehouse and returned to the depot. The pick
pallets are at random locations in a rectangular 250 x 200 feet warehouse with a middle
aisle and the depot is located at one end of the middle aisle. The travel speed of the
fork lift is assumed to be 350 ft/min when carrying 0, 1, or 2 pallets throughout the
experiment. A set of 10 different problem instances, in each pick and store locations are
randomly generating, was solved with the MILP model. Stackability density is set to
50%; i.e. only half of the possible stackable combinations are enabled. Table 4.2 lists
the objective value, solve time, and number of multi-command operations used in each
problem instance.

<table>
<thead>
<tr>
<th>Instance</th>
<th>Objective value (min)</th>
<th>Solve time (s)</th>
<th>Multi-command operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>22.40</td>
<td>3.81</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>22.93</td>
<td>3.35</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>24.69</td>
<td>3.82</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>22.90</td>
<td>3.63</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>23.59</td>
<td>4.07</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>25.37</td>
<td>3.13</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>24.16</td>
<td>3.49</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>25.54</td>
<td>3.75</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>24.60</td>
<td>3.99</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>23.99</td>
<td>3.94</td>
<td>4</td>
</tr>
<tr>
<td>Average</td>
<td>24.02</td>
<td>3.70</td>
<td>3.2</td>
</tr>
</tbody>
</table>

Table 3.1: Total route time for 20 location problems

The data shows that the objective value, the solve time, and the number of multi-
command operations vary for each problem instance. Distances between each store and
pick location change in each problem instance and therefore routing decisions and out-
put parameters change. To study the sensitivity of the problem output parameters are averaged.

The impact of various parameters of the problem such as pallet stackable density, vehicle travel speeds, pallet handling times, etc. on the routing decisions is now explored.

### 3.4.3 Pallet stackability density

Our intuition suggested that the total time required to put away and pick pallets would decrease, perhaps substantially, as the stackability density increases. If the stackability density in this 9 pallet problem in Section 3.4.1 is 100% (i.e., all pallets are stackable on each other), the total time is 11.14 minutes. This seemed a rather modest improvement when compared to the 11.85 minutes required for a stackability density of 12.5% so the numerical example is now used to test the sensitivity of the solution to changes in various model parameters.

The stackability density is varied from 0% to 100% in 10% increments for each of the 10 problem instances used in Section 3.4.2. This idea is common in many areas including facility layout problems. To achieve effective utilization of floor space pallet stacking has been proposed Gu et al. (2010) [54]. It was assumed that the increment complements the previous stackability density. Figure 3.3a shows the minimum total route time versus stackable density. It should be noted that a trend line has been added to show the trend of total time variation. Figure 3.3b shows the variation of computation time and number of multi-command operations when stackability density is varied for the same set of problems.

From the results in Figure 3.3a, it is clear that increasing pallet stackable density results in shorter total route time. It should be noted that when the pallet stackable density exceeds 50% of the total possible stacking combinations, total route time reduction is not significant. The total route time decreases by 3.76% when pallet stackable density increases from 0% to 50%; the decrease when pallet stackable density is increased from 50% to 100% is 0.73%. This is directly related to the number of multi-command operations,
which is shown in Figure 3.3b. Even though the number of multi-command operations increases when pallet stackable density is increased, it does not increase significantly when pallet stackable density exceeds 50%. The average number of multi-command operations increases from 0 to 3.2 when pallet stackable density is increased to 50%; however when the pallet stackable density is increased to 100%, the average number of multi-command operations increases to just 3.4, from previous 3.2.
The model calculates the routes that give minimum total time for each problem. The routes are a combination of single command, dual command and multi command operations that fulfil the store and pick requirements. When the pallet stackable density is 0%, no-pallet can be stacked which forces all single command operations. When pallets can be stacked on each other, the routes may include multi-command operations. Multi-command operations might save total time, depending on the store and pick locations of pallets, especially if the pallets are located far away from the depot and pallets are located closer to each other. However, single-command or dual-command operations might be efficient for handling some pallets, for example if pallets are closer to the depot. As a result, pallet stackability of them does not result in multi-command operations nor affect the objective value. Thus, pallet stackable density increase does not significantly reduce the objective function value after certain point.

Note also that as the pallet stackable density increases, the model is burdened with more possible multi-command operations. This is evident by the increase of computation time with increase of pallet stackable density in Figure 3.3b. Computation time increase is 35% when pallet stackable density is increased from 0% to 50%, while it becomes 80% when pallet stackable density is increased to 100% from 50%.

3.4.4 Vehicle travel speed

Fork truck lift travel speed is also an important factor when considering multi-command operations because single pick operations become more time-effective for pallets further from the depot as speed increases. This impact is explored using the same problem above - 10 pallets to be picked, 10 pallets to be stored, a 250x200 feet warehouse with one middle aisle.

In the first scenario, the average speed of the fork truck is varied between 3 and 8 mph (350 ft/min to 705 ft/min) in intervals of 1 mph. Again, the minimum total time is determined for 10 replications of each speed level for three pallet stackable densities: 0%, 50% and 100%. Figure 3.4a shows the minimum total route time for the different
scenarios while Figure 3.4b shows the average number of multi-command operations.

(a) Total route time vs. stackable density

![Graph showing total route time vs. stackable density.](image)

(b) Average number of multi-command operations vs. vehicle Speed

![Graph showing average number of multi-command operations vs. vehicle speed.](image)

Figure 3.4: Vehicle speed variation

Figure 3.4a confirms the obvious, that higher vehicle speeds result in shorter total route time for a given pallet stackable density. For instance, when stackable density is 50% total route time decreases by 32.7% when vehicle speed is increased to 8 mph from
3 mph. In addition, it is also clear that increasing pallet stackable density results in reduced total route times for slower vehicle speeds; however it does not improve total route time for higher vehicle speeds. The total route time reduction for 3 mph, 4 mph, and 5 mph travel speeds is 7.2%, 4.1%, and 2.3% respectively when stackable density increases from 0% to 100%. However, for higher speeds such as 6 mph, 7 mph, and 8 mph the total time reduction is 1.2%, 0.7%, and 0.5% respectively. Therefore it is evident that when the vehicle speed is slower than 6 mph, multi-command operations are sensitive to pallet stackable density variations while faster speeds are not sensitive to stackable density variations. Multi-command operations increase pallet handling time, however it reduces travel time. When the vehicles in the warehouse can travel fast, multi-command operations become less favorable. In other words, when travel time is the major contributor to the total time, multi-command operations becomes favorable. Figure 3.4b shows that average number of multi-command operations keeps decreasing when vehicle speed is increased, regardless of the pallet stackable density if pallets are stackable at all.

### 3.4.5 Asymmetric vehicle travel speeds

In practice, fork truck drivers may travel in reverse when pallets are stacked to have visibility. This typically demands, because of company policy or law, that the speed of the fork truck be reduced. To investigate how multi-command decisions are affected when stacked pallets force reduced speed of the fork truck, the average truck speed associated with carrying two stacked pallets ($s_b$) is varied from 100% to 50% of the full ($s_f$) speed in intervals of 10%. In these cases, the fork truck is allowed to operate at full speed when carrying zero or one pallet. As before, 10 randomly generated instances of each problem were solved. Figure 3.5a shows optimal total pallet handling time when speed is reduced as a result of carrying 2 pallets. Figure 3.5b shows average number of multi-command operations when speed is reduces as a result of carrying 2 pallets. We assumed that the pallet stackable density is 50%.

From Figure 3.5a, it is observable that for a given forward travel speed total
operation time increases when stacked travel speed reduces. Lowest total time is achieved when stacked travel speed is equal to the forward travel speed, i.e. 100% stacked travel speed. This is apparent for lower speeds. For instance, when stacked travel speed is 50% of the forward speed of 4 mph, total route time increases by 3.7%; the total route time increase when forward speed is 8 mph is 0.4%. Figure 3.5b shows that for a given
forward speed, desirability of using multi-command operations decreases when loaded speed reduces. When the vehicle speed is 4 mph, the average number of multi-command operations decrease from 3.2 to 0.2 when loaded travel speed becomes half of the full speed. With higher reduced speed, attractiveness of multi-command operations is further reduced. Average number of multi-command operations becomes 0 when loaded travel speed is 60% of a forward speed of 6 mph and it becomes 0 when loaded travel speed is 80% of a forward speed of 8 mph. Even when loads reduce speed, multi-command operations are preferred when the full speed of the truck is slower.

3.4.6 Pallet handling times

The next parameter investigated is pallet handling times that can vary from warehouse to warehouse based characteristics like equipment and width of aisles. Regardless of reason, handling times will certainly affect the effectiveness of multi-command operations. There are four pallet handling times, namely single pallet pick-up time \( t_s \), single pallet store time \( t_p \), pick up a pallet and stack on another \( t_{ps} \) and store a pallet from a stacked setup \( t_{ss} \). While many different experiments can be conducted with different pallet handling times, the following experiment demonstrates how stacked pallet handling times effects multi-command decisions.

In the previous example, it was assumed that single pallet handling times are equal \( t_s = t_p \) and stacked pallet handling times are equal \( t_{ps} = t_{ss} \). In this sensitivity analysis, the first perturbation is varying single pallet handling times \( (t_s, t_p) \) from 0.1 minutes to 0.5 minutes in 0.1 minute intervals. In all cases with which we are familiar, stacked pallet handling generally requires more time than single pallet handling. In this study, it is assumed that whatever reason causes single pallet handling to be faster or slower would also cause stacked handling to be faster or slower. Further, it is assumed that stacked pallet handling takes an additional amount of time that is a constant bias. Hence, stacked pallet handling times are varied as \( t_{ps} = t_{ss} = t_s + k \) where \( k \) varies from 0 to 0.5 minutes in 0.1 minute intervals. Experiment was conducted for a 250x200 feet
warehouse with one middle aisle where 10 pallets are to be picked and 10 pallets are to be stored and pallet stackable density is 50% to find total minimum time for 10 test problems.

(a) Total route time vs. pallet handling time difference

(b) Number of multi-command operations vs. pallet handling time difference

Figure 3.6: Pallet handling time variation

Figure 3.6a shows the total route time for different pallet handling times and Figure
3.6b shows average number of multi-command operations for different pallet handling times. Obviously, the total route time increases with each single pallet handling time. As intuition suggests, when stacked pallet handling time is equal to single pallet handling time, the optimal solution has multi-command operations. As a result, minimum route time for a given single pallet handling time is observed when stacked pallet handling time is equal to the single pallet handling time.

When stacked pallet handling takes more time compared to single pallet handling time, single-command operations are preferred over multi-command operations as evident from Figure 3.6a. Fewer multi-command operations result in longer travel and handling times which causes higher total route times. This trend is consistent throughout the single container handling time range. This suggests that the difference between the single pallet handling time and stacked container handling time is more important than single container handling time alone towards deciding on multi-command operations.

3.4.7 Warehouse size

Warehouses come in many different sizes and layouts. We considered similar problem to above with 20 pallets: 10 pallets to be stored and 10 pallets to be picked and returned back to the depot. Three warehouse sizes were considered to study the effect of warehouse size on the decision making process: Small (200 x 150 ft.), Medium (250 x 200 ft.) and Large (400 x 200 ft.). Figure 3.7a shows the total route time versus warehouse size and Figure 3.7b shows the average number of multi-command operations when warehouse size is varied. It was assumed that the pallet stackable density remains at 50% and vehicle speed remains the same regardless of number of pallets it carries.

The data in Figure 3.7b confirms the claim that lower speeds increases attractiveness of multi-command operations for a given warehouse size. This is reflected in the total route time as well, as evident in Figure 3.7a where slower speeds result in smaller total route times for a given warehouse size. Furthermore, for a given vehicle speed more multi-command operations are used when the warehouse size is increased. Total route
time, too, increases with the warehouse size.

Larger warehouse means the pallets are positioned farther away from each other and hence the travel time is higher compared to that of a smaller warehouse. This directly results in longer total route time in larger warehouses. However, considerable amount of
travel time can be reduced if multi-command operations are performed. Thus, larger warehouses prefer multi-command operations if pallet stacking is allowed. Data shows that the most number of multi-command operations were used in the largest warehouse and when the vehicle speed is slowest. Therefore, multi-command operations will be most efficient if used for larger warehouses with slow moving vehicles.

3.4.8 Warehouse aisle layout

There are various aisle layouts used in warehouses. We studied the effect of aisle layout on total travel time. We assumed that pallet stackable density remains at 50% and average forklift is 350 ft/min. We considered three aisle layouts in a 250 x 200 ft warehouse. Figure 3.8 shows the considered warehouse layouts, namely Middle aisle layout, Side aisles layout and Vertical aisle layout. The problems solved in this scenario are similar to the above: where 20 pallets located at random locations in the warehouse and 10 pallets are to be stored and 10 pallets are to be picked.

Figure 3.8: Warehouse aisle layouts

Figure 3.9 shows the variation of average number of multi-command operations, total route time, and solves time with the aisle layout. It should be noted that the desirability of multi-command operations is lower for a warehouse with vertical aisle layout. Compared to a warehouse with middle aisle design, travel distance between locations at the end of access aisles is lower in warehouses with side-aisles and vertical aisles. As a result, travel time reduces for multi-command operations and total route time reduces. Due
to lower travel distances, multi-command operations are less desirable for routes. Also, warehouse layout affects problem complexity, as indicated by the varying solve time.

### 3.4.9 Number of pallets

Since the problem is NP-hard, longer solve times should be expected for instances with higher number of pallets. The effect of solve time and desirability of multi-command operations when number of handled pallets is increased is studied. We considered problem instances with 20, 30 and 40 pallets in a 250 x 200 ft. warehouse with a middle aisle. We assumed that pallet stackable density remains at 50% and average forklift is 350 ft/min. Pick to store ratio was considered 1:1 for all instances. 10 problem instances for each number of pallets was solved using CPLEX model.

The data in Figure 3.10 confirms that the solve time increases exponentially when number of pallets is increased. Also, as intuition suggests, number of multi-command operations increases linearly with number of pallets. Since pallet stackable density remains at 50% and vehicle speed does not vary, number of multi-command operations is proportional to the number of pallets stored and picked.
A constrained optimization approach where some variables are considered hard constraints can be used to obtain practical solutions for realistic-size problems using the above mathematical model. Although this limits using some activity paths that can save route time, limited operational scope can dramatically reduces computational time and complexity although they are [0,1] models. We consider three models with different constraints for decision variables where all models consider stacking as an option while transporting pallets.

These models determine the routes for picking or storing \( n \) pallets. The assumptions are as before: all routes begin and end at the depot, fixed storage policy, and a one-to-one mapping between locations and pallets, maximum stacking of two pallets, depot is location 1 and the \( n \) pallets/locations are 2 through \( n + 1 \).
3.5.1 Method 1 - Separate store and pick

In this method, the routes for pallets to be stored and picked are determined separately; pallets can be stacked while transporting. A solution includes only $X_{ij}, X_i, X_{kl}$, and $X_{kl}$ decision variables while hard constraints set all the other variables to 0. This model is really single-command pallet handling operations where pallet stacking is possible as an option.

3.5.2 Method 2 - Separate store and pick with dual-command

In this method, a limited number of dual-command operations are allowed. In particular, a route with both store and pick activities are permitted along with single pallet store and pick of Method 1. That is, the forklift can follow one of the following three activity paths in a single route: 1) store one or two pallets, 2) store one pallet and pick one pallet, or 3) pick one or two pallets. In addition to above decision variables, $X_{ij}$ can be included in the solutions while all other decision variables are held at 0. Total number of pallets stored and/or picked in a single trip is limited to 2 in this method.

3.5.3 Method 3 - Limited multi-commands

This method allows a single route to include both store and pick activities and may handle more than two pallets in a single trip; however, storage and picking activities cannot alternate. That is, all storing operations need to be finished before picking operations start. In the model, $X_{ikj}, X_{ikjl}, X_{ki}$, and $X_{ki}$ are set to 0 while all other decision variables are used to calculate routes.

3.5.4 Numerical Examples for constrained optimization

The example depicted in Figure 3.2 that contains 9 pallets is first solved to compare the constrained optimization methods. Recall that pallets 1 through 4 must be stored
while pallets 5 through 9 are to be picked. The comparisons shown in Table 3.2 includes the total time to store and pick the pallets and computation time to obtain the solution.

<table>
<thead>
<tr>
<th>Time</th>
<th>MIP</th>
<th>Heuristic 1</th>
<th>Heuristic 2</th>
<th>Heuristic 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Route Time (min)</td>
<td>11.14</td>
<td>14.51</td>
<td>12.86</td>
<td>11.14</td>
</tr>
<tr>
<td>Computing Time (sec)</td>
<td>2.59</td>
<td>1.72</td>
<td>1.19</td>
<td>2.01</td>
</tr>
</tbody>
</table>

Table 3.2: Comparisons of heuristic results for the 9 pallet problem

As expected, Method 1 requires the least computing time but the solution is 30% above the optimal even for this simple problem. The solution from Method 2 is only 15% above optimal which would likely be acceptable in practice and the computing time is reduced over 30%. The simple structure of this example allowed Method 3 to find the optimal solution in 20% less time. This would certainly not always be the case when the optimal solution includes a number of alternating store and picks; however, it could be a very good heuristic in practice.

<table>
<thead>
<tr>
<th>No. of pallets</th>
<th>Exact</th>
<th>Single</th>
<th>Method 1</th>
<th>Method 2</th>
<th>Method 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z (min)</td>
<td>Z (min)</td>
<td>CPU (s)</td>
<td>Z (min)</td>
<td>CPU (s)</td>
<td>Gap(%)</td>
</tr>
<tr>
<td>20</td>
<td>23.84</td>
<td>3.81</td>
<td>31.57</td>
<td>28.32</td>
<td>0.79</td>
</tr>
<tr>
<td>50</td>
<td>57.91</td>
<td>516.42</td>
<td>79.27</td>
<td>69.84</td>
<td>1.24</td>
</tr>
<tr>
<td>100</td>
<td>–</td>
<td>176.85</td>
<td>145.47</td>
<td>524.80</td>
<td>–</td>
</tr>
</tbody>
</table>

Table 3.3: Comparison of total time and computing time for optimal method and constrained optimization methods

The above methods are used to find solutions to problem instances with 20, 50, and 100 pallets where half of the pallets are stored and rest of them are picked. Each problem has a stackability density of 50% and 10 replications or each are solved for 20, 50 and 100 pallets. Table 3.3 summarizes total times store and pick pallets and computation times. Optimality gap -defined as: (heuristic solution - optimal solution)/optimal solution - is also given. Objective time for single command operations where single pallet is stored or picked in each trip is also given for comparison. It should be noted that objective time and computation time for Method 3 in 100 pallet example is averaged for only 2 test problems.

The solution found by Methods 1 and 2 are very fast compared with the time to find
an optimal solution and Method 3. This gets more pronounced as the number of pallets increases. Achieving fast solutions is a result of having a relatively small search space in Methods 1 and 2. In both examples where the optimal solution could be found, the optimality gap is around 20%. The optimality gap for the solutions to the 20 and 50 pallet problems using Method 2 is about 4%. For practical implementation this heuristic could be attractive especially considering the dramatic reduction in computing time. When the operating restrictions in Method 2 are compared with the lack of restrictions in the optimization model, the difference is that alternating picks and stores are not allowed in Method 2. These results suggest that as the number of pallets increases and the stackability density is rather high, the time saving associated with alternating picks and stores is small.

Method 3 is the most complex with fewer operating restrictions; hence, it produces the best results. The better results must be balanced with the computation times which are high compared to the other two methods but lower than the MILP method. This method can be used to find near optimal results with shorter computation time than that of MILP method. The resulting routes can be less complex and easier to execute than optimal routes from MILP method. However, it should be noted that this method gets prohibitively expensive with computing time when problem size increases. This method was not able to solve some of the 100 pallet problems because of the problem size. Therefore, for larger problem instances, Method 3 might not be applicable similar to MILP method.

These limited numerical examples are not intended to provide recommendations on which methods are superior to the others or that certain heuristics are superior for warehouses with specific characteristic. Rather, these have been provided to illustrate an interesting idea of using simpler optimization models that restrict operating options as heuristics for realistically sized instances of this complex problem. The preliminary conclusion is that this approach holds promise for finding very good solutions in a fraction of the time it takes to find the optimal solution if it can be found at all.
3.6 Construction Heuristics

Most published research for the VRP has focused on the development of heuristics due to computational requirements for large size problems. In this research, too, heuristic methods are required in order to solve practical size problems. However, the special constraints in this problem, namely limited vehicle capacity and stackability constraints, makes popular heuristic approaches less useful without extensive modifications. We study several construction heuristics to solve realistic size problem instance in this paper. Construction heuristics were selected for their ability to provide solutions quickly.

3.6.1 Heuristic 1: Separate store and pick

Routes are constructed by visiting each location until all store and pick requests are fulfilled. Pallets are stacked if vehicle capacity and stackability restrictions permits. Pick operations are not started until all store operations are fulfilled. The steps of the separate store and pick heuristic are given as follows.

**Step 1.** Assign a unique rank for each location randomly.

**Step 2.** Visit the lowest ranked location with unfulfilled store demand \((i)\) and visit.

**Step 3.** Find the closest location to \(i\) with unfulfilled store demand that can be stacked; if available, visit; return to the depot.

**Step 4.** Return to Step 2 until all the store locations are visited.

**Step 5.** Visit the lowest ranked location with unfulfilled pick demand\((j)\) and visit.

**Step 6.** Find the closest location to \(j\) with unfulfilled pick demand that can be stacked; if available, visit; return to the depot.

**Step 7.** Return to Step 5 until all the pick locations are visited.

**Step 8.** Calculate Total time \(T\) for the routes.

**Step 9.** If Total time \(T\) is less than \(T_{min}\) update \(BestRoute\).

**Step 10.** If the stopping criterion (number of iterations) is satisfied, output \(BestRoute\) and \(T_{min}\); otherwise return to Step 1.
At the beginning of each iteration, a random visiting rank is assigned to each location. After creating separate store and pick routes, total route time for the entire route is calculated and compared with the the total time of previous best route. If the total time of the current iteration is smaller than previous best route, current route is selected as the best route. This is repeated until the stopping criterion is met, which is the maximum number of iterations. Detailed pseudo code for the separate store and pick heuristic is given in Algorithm 1 in Appendix A.

3.6.2 Heuristic 2: Dual command operations

This heuristic generates routes based on dual command operations. That is, a single trip can store and pick pallets in this construction heuristic; however pallets are not stacked while transported. The steps of the dual command operations heuristic are as follows.

**Step 1.** Assign a unique rank for each location randomly.

**Step 2.** Visit the lowest ranked location with unfulfilled store demand \((i)\) and visit.

**Step 3.** Find the closest location to \(i\) with unfulfilled pick demand; if available visit; return to the depot.

**Step 4.** Return to Step 2 until all the store locations are visited.

**Step 5.** Visit the lowest ranked location with unfulfilled pick demand \((j)\) and visit; return to the depot.

**Step 6.** Return to Step 5 until all the pick locations are visited.

**Step 7.** Calculate Total time \(T\) for the routes.

**Step 8.** If Total time \(T\) is less than \(T_{min}\) update BestRoute.

**Step 9.** If the stopping criterion (number of iterations) is satisfied, output BestRoute and \(T_{min}\); otherwise return to Step 1.

Similar to heuristic 1, a random visiting rank is assigned to each location at the beginning of each iteration. Routes are constructed by visiting location with lowest rank with store demand and then visiting the closest location with pick demand and repeating until all store and pick demand is fulfilled. Similar to heuristic 1, best route is selected
by comparing the total route time. Stopping criteria is set with an upper bound for iterations. Detailed pseudo code for the dual command heuristic 2 is given in Algorithm 2 in Appendix A.

3.6.3 Heuristic 3: Multi-command operations

Limited multi-command operations based routes are generated in this heuristic. In each trip, storing operations need to be finished before performing pick-up operations. Pallet stacking is allowed during transportation. The steps of the multi-command operations heuristic are as follows.

**Step 1.** Assign a unique rank for each location randomly.

**Step 2.** Visit the lowest ranked location with unfulfilled store demand \((i)\) and visit.

**Step 3.** Find the closest location to \(i\) with unfulfilled store demand \((j)\) that can be stacked; if available visit; else go to Step 4 with \(j \leftarrow i\)

**Step 4.** Find the closest location to \(j\) with unfulfilled pick demand \((k)\); if available visit; else return to the depot.

**Step 5.** Find the closest location to \(k\) with unfulfilled pick demand that can be stacked; if available visit; else return to the depot.

**Step 6.** Return to Step 2 until all the store locations are visited.

**Step 7.** Visit the lowest ranked location with unfulfilled pick demand \((l)\) and visit

**Step 8.** Find the closest location to \(l\) with unfulfilled pick demand that can be stacked; if available visit; else return to the depot.

**Step 9.** Return to Step 7 until all the pick locations are visited.

**Step 10.** Calculate Total time \(T\) for the routes.

**Step 11.** If Total time \(T\) is less than \(T_{\text{min}}\) update BestRoute.

**Step 12.** If the stopping criterion (number of iterations) is satisfied, output BestRoute and \(T_{\text{min}}\); otherwise return to Step 1.

Routes starts with the lowest ranked store demand location, which is assigned at the beginning of each iteration. Then the closest location with a store demand and stackable is visited in the same route. Next location to visit is selected as the closest location
with pick demand. If there is a location with stackable pick demand, that is added to the route as well before returning to the depot. As before, total route time is compared to select the best route. After a set number of iterations, heuristic is stopped. Algorithm 3 in Appendix A shows the detailed pseudo code for the multi-command operations heuristic.

### 3.6.4 Numerical examples for heuristic methods

Problem instances with 20, 50, 100, and 200 pallets were solved using above heuristics where stackability density is 50% and number of pallets stored is half the total pallets. Minimum objective value and CPU time for each heuristic for 10 problem instances are averaged and presented in Table 3.4. It should be noted that the problem instances for 20 and 50 pallets use the same locations as in the constrained optimization methods in Section 3.5. The maximum number of iterations used for problems was $5n$ where $n$ is the number of locations. None of the other parameters were changed from the previous problems.

<table>
<thead>
<tr>
<th>Number of pallets</th>
<th>Exact</th>
<th>Heuristic 1</th>
<th>Heuristic 2</th>
<th>Heuristic 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$Z$ (min)</td>
<td>CPU (s)</td>
<td>$Z$ (min)</td>
<td>CPU (s)</td>
</tr>
<tr>
<td>20</td>
<td>23.84</td>
<td>3.81</td>
<td>26.51</td>
<td>0.04</td>
</tr>
<tr>
<td>50</td>
<td>57.91</td>
<td>516.42</td>
<td>67.41</td>
<td>0.30</td>
</tr>
<tr>
<td>100</td>
<td>–</td>
<td>–</td>
<td>141.49</td>
<td>2.06</td>
</tr>
</tbody>
</table>

Table 3.4: Comparison of total time and computing time for optimal method and heuristic methods

From the Table 3.4, it can be noted that all three heuristics provide solutions much quicker than the exact approach. For these example problems, there was no statistical difference between the computing times of the three heuristics. The heuristics are able to solve the problems with large number of pallets within a short computing time. Objective times provides by heuristic 1 is the highest compared to other heuristics. This suggests that constructing separate routes for store and pick operations is inefficient. Heuristic 2 constructs routes with approximately 5% optimality gap and routes by heuristic 3 have approximately 8% optimality gap.
For the problem instances with exact optimum solution where number of pallets is 20 and 50, best solutions were given by heuristic 2 which is the dual-command operations. However, when number of pallets is increased to 100 and 200, heuristic 3 where pallet stacking is prioritized provided better solutions over heuristic 2 which is dual command operations. When number of pallets is low in a problem instance, the pallets are scattered around the warehouse. In such situations, handling stacked pallets is not worth because traveling between locations is a major portion of the total time. However, when number of pallets is increased and pallet density per area is higher, pallet handling time is a significant portion of the total time. Hence, using stacked pallets during travel is beneficial because it reduces travel time between the depot and locations. As a result, heuristic 3 gives better routes with smaller objective value when the number of pallets is high.

<table>
<thead>
<tr>
<th>Number of pallets</th>
<th>Exact</th>
<th>Constrained Optimization Method 2</th>
<th>Best Heuristic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Z (min)</td>
<td>CPU (s)</td>
<td>Z (min)</td>
</tr>
<tr>
<td>20</td>
<td>23.84</td>
<td>3.81</td>
<td>24.86</td>
</tr>
<tr>
<td>50</td>
<td>57.91</td>
<td>516.42</td>
<td>60.12</td>
</tr>
<tr>
<td>100</td>
<td>–</td>
<td>–</td>
<td>126.84</td>
</tr>
<tr>
<td>200</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>

Table 3.5: Comparison of total time and computing time for best methods

Table 3.5 illustrates the comparison between the exact solution, the best heuristic, and Method 2 in the constrained optimization approach. Recall that Method 2 was the approach that seemed to exhibit a nice balance between solution quality and computing time. It is clear that both heuristics and the constrained approaches are capable of finding a solution to much larger problem instances where exact methods cannot determine a feasible solution. If constrained approaches require more computing time and resources than are available, heuristics would appear to be a good alternative. It should be noted that the constrained optimization methods was unable to solve 200 pallets problem instances.

An interesting observation is that heuristic 3 is able to construct routes with smaller objective than that from Method 2 in a substantially smaller computing time. Multi-command operations with stacking in heuristic 3 is able to provide better results compared to separate store and pick operations with dual-command operations by Method 2. This
shows the importance of multi-command operations in unit load handling.

While the above heuristic approaches do provide feasible solutions quickly, there is opportunity to improve the solutions further. Future research can focus on metaheuristic methods that provide better approximations to the optimal solution. This would be particularly important in a large warehouse with multiple material handling devices.

### 3.6.5 Location visiting order for heuristic methods

Each heuristic described in this section uses a random location visiting order from the depot. The sensitivity to different location visiting orders is explored here. Instead of visiting the lowest ranked location,(the rank for each location was assigned randomly in heuristics) locations are visited based on the distance from the depot. In the first scenario, each route is started with location that is closest to the depot with unfulfilled store or pick demand. In the second scenario, the location that is located farthest from the depot with unfulfilled store or pick demand is visited first in each route. Since routes are generated in either ascending or descending order of distance from the depot, and distance between each location is fixed only one iteration is required. Table 3.6 shows the average total route time, computation time for the different location visiting orders for averaged over 10 instances of 20 pallet problem.

<table>
<thead>
<tr>
<th>Heuristic</th>
<th>Closest location Z (min)</th>
<th>Closest location CPU (ms)</th>
<th>Farthest location Z (min)</th>
<th>Farthest location CPU (ms)</th>
<th>Random location Z (min)</th>
<th>Random location CPU (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>28.25</td>
<td>10.55</td>
<td>27.90</td>
<td>7.81</td>
<td>26.51</td>
<td>39.84</td>
</tr>
<tr>
<td>2</td>
<td>25.60</td>
<td>2.34</td>
<td>25.39</td>
<td>2.34</td>
<td>24.81</td>
<td>33.20</td>
</tr>
<tr>
<td>3</td>
<td>26.50</td>
<td>1.17</td>
<td>27.32</td>
<td>0.78</td>
<td>25.50</td>
<td>30.47</td>
</tr>
</tbody>
</table>

Table 3.6: Comparison of location visiting order in heuristic methods

The data shows that random location visit constructs routes with minimum total time. However, the computation time for random location visit method is several times higher comparatively due to many iterations required compared to one iteration in distance based location visit order methods. Since the computation times are in milliseconds range, constructing routes with use of several iterations can be justified.
3.7 Conclusions and Future Work

This research has explored warehouse operations in which forklifts can store and pick multiple pallets on one route by stacking them. The mixed integer linear programming model was developed and solved for smaller problem instances which allowed the sensitive of the objective function and run time to some parameters were investigated. The underlying fact that the problem belongs to the computation complexity class NP-hard; however, restricts the number of pallets that can be used and an optimal solution obtained. Therefore 3 constrained optimization methods were proposed which provides sub-optimal routes with shorter computation time. Furthermore three heuristic approaches using a construction heuristics were introduced to find solutions to larger problems very quickly.

In a realistic sized unit load warehouse application where hundreds of pallets need to be stored and picked during a work shift, determining routes using the mixed integer linear programming model would be practically impossible, even with the constrained optimization methods. In addition, the assumption of a static problem where store and pick list does not change during the routing operation might not be valid in some applications. Store and pick list might need to be updated several times during a work shift due to reasons such as priorities, interruptions, etc. A practical approach would be to break the store and pick demand into smaller batches and use the optimization method to find routes. Since the mathematical model is able to find optimal routes for smaller problems within few minutes, it is possible to solve the problem using a rolling horizon decision making approach and update the store and pick demand list accordingly. Alternatively, realistic sized problems may be solved using all three proposed heuristics and choose the best set of routes because all three heuristics are able to solve large problem instances within a minute of the computing time.

Since this problem is only just beginning to be addressed, there is much work that can be done. Using local search heuristics to improve the sub-optimal solutions by constrained optimization methods and heuristics methods is an immediate future research.
Applying evolutionary algorithm or metaheuristics to find better solutions is one well-worn path but maybe there are more interesting and fruitful avenues. For example, in a warehouse using random storage, the pallets to be picked have a storage location but the ones to be stored are free to be put in any free location. In this case, the model could be altered to determine the minimum time storing and picking routes as well as the storage locations for the pallets to be stored. Taking this one step further, many distribution centers know the inbound freight scheduled for delivery and the orders that must be filled, at least for the foreseeable future. Modifying the model for the random storage case with the details of several future orders to be picked known could add significantly to the value of the results as well insight for good heuristics.
Chapter 4

Multi-Command Routing Operations in Unit Load Storage Facilities

4.1 Introduction

Warehouse storage space is a limited resource and therefore using space efficient methods to store pallets is a common goal for warehouse management. Aisle space is considered as non-revenue generating space and therefore, aisle width reduction is considered as a measure of increasing the space utilization. However, narrow aisles result in one way travel through aisles and possible congestion and increased travel distance and travel time.

To use the available space efficiently, warehouses store more than one pallet in an aisle location. The simplest way of storing pallets is on floor, typically arranged in lanes. In addition, pallets may be stored on each other or on supporting racks. Thus, at a single location which can be accessed by a vehicle, multiple pallets may be stored. Ability to store more than one pallet at a location makes pick and store operation decisions complex because multiple pallets can be stored/ picked in a single visit to a location. In this scenario, locations and unit loads do not have a one-to-one mapping relationship.

When there is one-to-one mapping relationship between pallets and locations, and
not more than two pallets are carried in a routing vehicle which was explored in Chapter 3, enumerating the possible pallet carrying combinations is effective for deciding the optimum routes. However, when a one-to-one mapping relationship is not available, enumerating is inefficient. In addition, the possibility to carry more than two unit loads in a vehicle should be considered for more practical applications. Hence, in this chapter we explore generalized multi-command operations.

4.2 Background

Generally, an order picking warehouse system serves as the base for pick and delivery operations. Among warehouse functions including receiving, storage, order picking and shipping, order picking is known to be the most labor intensive and costly one [45]. Picker-to-parts systems, in which order pickers walk or ride through the warehouse and collect the requested items, is an important part of manual order picking systems [126]. In the order-batching problem (OBP), in which customer orders are batched such that total travel distance of all pickers is minimized, the capacity of the picking device is assumed to be greater than the capacity of the pick demand of a location and the total demand of the warehouse is greater than the picking device capacity for the problem, to avoid the problem being infeasible or trivial, respectively. VRP based batching heuristics, namely seed algorithm and savings heuristics, are used to minimize total picking effort in problem of partitioning among pickers. De Koster et al. (1999) [38] evaluate batching and routing algorithms together in the picking process to form a single picking route. They evaluate both Seed algorithms and the Time Savings algorithms. Gu et al. (2007) [53] presents a comprehensive summary of work on batching and partitioning problems. Henn and Wäscher (2012) [63] propose Tabu Search and Attribute-based Hill Climber approach to solve OBP in manual order picking systems. Öncan (2013) [94] propose a Genetic Algorithm based solution for OBP. Öncan (2015) [95] introduce mixed integer linear programming formulations for OBP which considers traversal, return and midfield
routing policies and develop an Iterated Local Search Algorithm with Tabu Thresholding.

The OBP focuses on only one of the three operation level activities in manual warehouses: assignment of items, grouping of customer orders into batches, and routing of order pickers through the warehouse. However, OBP does not consider simultaneous item storing and picking. Vehicle Routing Problems with Backhauls (VRPB) focuses on goods transportation between locations where goods can be either delivered to a destination or picked from a destination.

Parragh et al. (2007) identifies four subclasses of VRPB in items are transported from and to a depot. First subclass is when all the delivery location are visited before visiting the first pick location and is identified as Vehicle Routing Problem with Clustered Backhauls (VRPCB). Second subclass allows mixed visiting of delivery and pick locations. This problem class is identified as VRP with Mixed linehauls and Backhauls (VRPMB) and is also known as Mixed Vehicle Routing Problem with Backhauls (MVRPB) and Vehicle Routing Problem with Backhauls with Mixed load (VRPBM). It should be noted that in above two problem subclasses locations are either delivery or pick customers but cannot be both.

In the third subclass each location has pick and delivery demand and each location is allowed to be visited once for a combined pick and delivery, or twice if these two operations are performed separately. First, few locations are visited to partially empty the vehicle and then locations are visited to deliver and pick. Picks for the first visited locations are done at the end. This problem is identified as VRP with Divisible Delivery and Pickup (VRPDDP). Gribkovskaia et al. (2007) studied the single vehicle version namely Single Vehicle Routing Problem with Picks and Deliveries (SVRPPD). Classical construction and improvement heuristics, as well as a tabu search heuristic are used to solve for the problem. Archetti et al. (2006) studied the above problem with splitting of the delivery or the pick allowed also known as the Split Delivery VRP(SDVRP). In SDVRP, the demand of each location can be greater than the capacity of vehicles, and each location can be visited more than once by a fleet of homogeneous vehicles. Tabu
search algorithm is proposed to find a set of routes such that travel distance is minimized. Archetti et al. (2012) [21] surveys the SDVRP and present formulation, properties and solution approaches. However, SDVRP does not consider simultaneous pick while visiting locations.

Problems generally known as VRP with simultaneous pick and delivery (VRPSPD) in which every location is associated with a store as well as a pick quantity and each location can only be visited exactly once is addressed in the fourth subclass. Deliveries are started from a single depot at the beginning of the each route and picked loads are brought back to the same depot at the end of each route. Nagy and Salhi (2005) [90] propose heuristic methods for the VRPSPD when goods can be picked up before all deliveries are made, not traditional delivery-first pick-second problems. This heuristics is capable of solving problems with multiple depots after appropriate modifications. Montané and Galvao (2006) [88] develop a tabu search algorithm to solve the VRPSPD. Fard an Akbari (2013) [46] propose a hybrid tabu search algorithm for VRPSPD with maximum tour time length.

The aforementioned literature provides base for this research; This research combines the split delivery characteristic of SDVRP with simultaneous pick and delivery characteristic from VRPSPD. In addition time spent at the location is taken into consideration for routing decisions.

4.3 Multi-Command Optimization Model

In this research, we investigate methods for determining routes that minimize total route time in a warehouse based without one-to-one mapping relationship between unit loads and locations and vehicle capacity is not limited to two pallets.
4.3.1 Problem definition

Here the warehouse is represented as a directed network. Unit load locations are a set of vertices and the set of edges connecting the vertices represent the paths between each location. A single vehicle traverses this network and it incurs cost to travel each edge, which is equal to the time to travel distance between the vertices connected by the edge. At a vertex, a cost is incurred, which is equal to the time to handle the unit load(s). At the origin, the vehicle may pick some of the unit loads that need to be stored, and the number of unit loads picked is subject to the vehicle’s capacity limitation. At a location, the vehicle stores all unit loads terminating at the location and may pick unit loads need to be picked; thus, a location can be an origin and a destination simultaneously. A route is defined as the sequence of origins and destinations that the vehicle follows in storing and picking unit loads. The cost of a route is equivalent to the total time to travel and handle unit loads while completing the store and pick operations. The objective of the problem is to minimize the total cost of all the routes.

A key characteristic of the warehouse is the absence of one-to-one relationship between locations and unit loads. In contrast to one-to-one mapping between location and unit load characteristic in Chapter 3, this problem assumes that more than one unit load can be stored at a location. As a result, time to handle pallets at destinations is a significant portion of total route time. Therefore, it is important to consider total route time, i.e. travel time and unit load handle time, instead of just the total distance traveled. That is, when handling time can be ignored, minimizing distance and time are equivalent. When handling cannot be ignored like this problem, they are not.

It is assumed that that a single unit load handling vehicle carries out all store and pick operations. The unit load handling vehicle can carry more than one unit load during movement and it has a maximum capacity of $Q$ unit loads. If the vehicle has a capacity higher than the maximum store and pick demand of each location, each location requires to be visited only once. However, in this research problem it is assumed that the maximum number of unit loads per location is greater than unit load handling vehicle
capacity and thus unit load handling vehicle might need multiple trips to complete the store and/or pick demand at some locations.

In Chapter 3, it was noted that unit loads can have different characteristics and only some of them can be stacked on another. This characteristic was named 'stackability', and was included into routing decisions. In this problem it is assumed that all unit loads are stackable on each other and stackability is 100%. This means any unit load can be grouped with any other without restrictions to be carried in the handling vehicle.

Time to store a single unit load is $t_s$ and $t_p$ is the time for picking a single unit load. When more than one unit load is stored or picked, handling time is not equivalent to sum of handling all unit loads individually. For example, the time to store two unit loads together is not $2 \times t_s$. It would take less time to store or pick stacked unit loads than storing or picking them individually. Moreover, it is assumed that the time savings from handling stacked unit loads increases non-linearly with number of unit loads handled. For example, time to store 4 unit loads in a single operation is less than time to store 2 unit loads twice. Time for storing $x$ unit loads is $x^k \cdot t_s$ where $k$ is a constant less than 1. Similarly, time for picking $x$ unit loads is $x^k \cdot t_p$.

The warehouse has fixed storage locations where the distance between each location including the depot is known. At the beginning of the time period, store and pick demand for each location is known. It is assumed that the store and pick demand does not change until all store and pick operations are performed. All unit loads to be stored are located at the depot and all picked unit loads are brought back to the depot.

The store and pick demands of all the locations must be satisfied, and the number of unit loads carried in each section of the route cannot exceed the vehicle capacity $Q$. The objective is to minimize the total route time of the vehicle.

### 4.3.2 Notation

Let $G = (V, E)$ be a directed graph, $V = \{0, 1, ..., n\}$, is the vertex set where vertex 0 denotes the depot while the other vertices are the customers and $E = (i, j) : i, j \in V, i \neq j$
is the edge set. An integer pick demand $p_j$ and store demand $s_j$ is associated with each location $j \in V - \{0\}$. A single vehicle with a capacity $Q$ can travel an unlimited number of routes to fulfill the store and pick demand. However, for practical applications an upper bound $m$ on the number of routes is set. For example, if the vehicle store or pick only one unit load in each trip, maximum number of routes is equivalent to the total store and pick demand ($m = \sum_{i=1}^{n} (s_i + p_j)$). If store and pick demand is satisfied in single-command operations where vehicle carries up to its capacity in separate store and pick routes to each location, maximum number of routes is $m = \sum_{i=1}^{n} (s_i/Q + p_i/Q)$.

The parameters and variables are as summarized below.

- $n =$ total number of customers; $n = |V| - 1$
- $d_{ij} =$ distance from location $i$ to $j$ in feet; $i, j = 0, ..., n$
- $s_j =$ store demand at location $j; j = 1, ..., n$ in unit loads
- $p_j =$ pick demand at location $j; j = 1, ..., n$ in unit loads
- $t_p =$ time to pick a single unit load and load into the vehicle in minutes
- $t_s =$ time to store a single unit load at a location in minutes
- $Q =$ unit load handling vehicle capacity in unit loads
- $v =$ unit load handling vehicle speed in feet per minute
- $m =$ maximum number of routes
- $k =$ multi-unit load savings factor; $k < 1$

**Decision variables**

- $x_{ij}^t = \begin{cases} 1, & \text{if arc } (i, j) \text{ belongs to the route } t \\ 0, & \text{otherwise} \end{cases}$
- $y_{ij}^t =$ number of unit loads carried in arc $(i, j)$ to be stored at locations after visiting node $i$ in route $t$
- $z_{ij}^t =$ number of unit loads already picked-up in the route from locations already visited and at node $i$ and carried in arc $(i, j)$ in route $t$
4.3.3 Mathematical Model

Using the above notation, the mixed integer programming formulation for minimum total time which is inspired by Montané and Galvao (2006) [88], is presented. It should be noted that the multi-unit load savings factor is used as a linear parameter to preserve linearity.

Minimize

\[
Z = \sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{t=1}^{m} \frac{d_{ij}}{v} x_{ij}^t + \sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{t=1}^{m} k(t_s y_{ij}^t + t_p z_{ij}^t) 
\]  

subject to:

\[
\sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{t=1}^{m} x_{ij}^t \geq 1 \quad j = 0, ..., n \]  

\[
\sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{t=1}^{m} x_{ip}^t - \sum_{j=0}^{n} \sum_{t=1}^{m} x_{pj}^t = 0 \quad p = 0, ..., n; t = 1, ..., m \]  

\[
\sum_{i \in S} \sum_{j \in S} \sum_{t=1}^{m} x_{ij}^t \leq |S| - 1 \quad t = 1, ..., m; S \subseteq V - \{0\} \]  

\[
\sum_{j=0}^{n} \sum_{t=1}^{m} x_{0j}^t \leq 1 \quad t = 1, ..., m \]  

\[
\sum_{i=0}^{n} \sum_{j=0}^{n} \sum_{t=1}^{m} y_{ij}^t - \sum_{k=0}^{n} \sum_{t=1}^{m} y_{jk}^t = s_j \quad j = 1, ..., n \]  

\[
\sum_{k=0}^{n} \sum_{t=1}^{m} z_{jk}^t - \sum_{i=0}^{n} \sum_{t=1}^{m} z_{ij}^t = p_j \quad j = 1, ..., n \]  

\[
y_{ij}^t + z_{ij}^t \leq Q x_{ij}^t \quad i = 0, ..., n; j = 0, ..., n; t = 1, ..., m \]  

\[
x_{ij}^t \in \{0, 1\} \quad i = 0, ..., n; j = 0, ..., n; t = 1, ..., m \]  

\[
y_{ij}^t \geq 0 \quad i = 0, ..., n; j = 0, ..., n; t = 1, ..., m \]  

\[
z_{ij}^t \geq 0 \quad i = 0, ..., n; j = 0, ..., n; t = 1, ..., m \]
Constraints (4.2)-(4.4) are the classical routing constraints. Constraint (4.2) impose that each location is visited at least once. (4.3) are the flow conservation constraints that guarantees that arrivals and departures from each location are conducted in the same route. (4.4) are the subtour elimination constraints. Constraints (4.5) limit the maximum number of trips that can be used. Constraints (4.6) and (4.7) are the flow equations for store and pick demand. (4.6) ensure that the store demand of each location is satisfied while (4.7) guarantee that the pick demand of each location is satisfied. Constraints (4.8) establish that store and pick unit loads will only be transported using arcs included in the solution; furthermore, they impose an upper limit on the total unit loads carried by a vehicle in any given section of the route. Constraints (4.9) - (4.11) define the nature of the decision variables.

Archetti et al. (2005) [20] showed that SDVRP with integer demands can be solved in polynomial time for $Q = 2$ while it is NP-hard for $Q \geq 3$. The current problem is a generalization of SDVRP (by equating pick demands of customers to zero, and equating time to store and pick a unit load to zero), and hence it is a NP-Hard problem too. Because of this, we should seek heuristic methods for solving it.

### 4.4 Heuristic Methods

A seed algorithm selects initially a single seed order in the batch. More orders are then added according to a route closeness criterion until no more orders can be added due to a capacity constraint. The capacity constraint can be based on total pick time, number of orders in the batch, or weight. A savings heuristic starts by assigning each order to a separate batch. The algorithm then iteratively selects a pair of batches to be combined based on the savings of combining them until no more batches can be combined due to the capacity constraint.
4.4.1 Random location visit route construction heuristic

The first heuristic is based on a common dual-command heuristic for multiple unit
load handling. Similar to the dual-command heuristic, all routes start at the depot. In the
random location visit heuristic, after completing the store & pick requests of a location,
another location with store or pick demand may be visited if the unit load handling vehicle
has excess capacity. The idea is to construct routes such that vehicle capacity is used as
much as possible. The steps of the separate store and pick heuristic are given as follows:

**Step 1.** Assign a unique rank for each location randomly.

**Step 2.** Select the lowest ranked location with unfulfilled store or pick demand (i). If
demand of all locations is fulfilled go to Step 7.

**Step 3.** Visit location i. Store and/or deliver maximum possible demand.

**Step 4.** If the store and pick demand of current location is fulfilled, calculate *Excess
vehicle capacity*; If it is > 0 go to Step 5; If it is = 0 return to the depot and go to
Step 2. Otherwise return to the depot and go to Step 3.

**Step 5.** Select the next lowest ranked location with unfulfilled store or pick demand (j).

**Step 6.** If the store or pick demand of location j can be fulfilled with *Excess vehicle
capacity* or exceeds vehicle capacity, set i ← j and go to Step 3. Otherwise return
to depot and go to Step 2.

**Step 7.** For each constructed route, rearrange visiting order by pairwise exchanging des-
tinations.

**Step 8.** Check feasibility for each rearranged route and calculate operating time. Choose
the set of routes with least time (BestRoute).

**Step 9.** Calculate Total time T for the routes.

**Step 10.** If Total time T is less than T_min update BestRoute.

**Step 11.** If the stopping criterion (number of iterations) is satisfied, output BestRoute
and T_min; otherwise return to Step 1.

In the Random location visit heuristic, the first step is to assign a unique rank to
each location randomly. This rank would not change throughout a single iteration, but
changes in each iteration. Each trip is started from the depot and next location to visit
is selected based on the rank. The lowest ranked location that has an unfulfilled store or
pick demand (i) is visited at the beginning of each trip. Once the location i is visited,
next location to visit is based on two factors: whether the store & pick demand of location of satisfied or not, and if satisfied does the vehicle has excess capacity to store or pick more unit loads. If the demand of the location $i$ is not satisfied, return to the depot and revisit the same location to fulfill remaining demand. Once the demand of the location $i$ is satisfied and if the vehicle does not have excess capacity, then the vehicle should return to the depot and find a new location to visit. If the vehicle has excess capacity after satisfying the demand at $i$, then the next visiting location $j$ is selected based on the rank. The location $j$ is visited only if the excess capacity can fulfill the store or pick demand of location $j$ or if the store or pick demand of location $j$ requires multiple visits. Otherwise, even if the vehicle has excess capacity after visiting location $i$, return to the depot and find the next location to visit based on the rank. Above steps are repeated by location until all pallets are stored and picked.

After constructing all routes, a local search is then employed by pairwise exchange of destinations in each trip to see if an improvement can be found. For example, if a constructed trip for a vehicle with 4 unit load capacity is [depot $\rightarrow$ Location 1 - store 2 & pick 1 $\rightarrow$ Location 2 - pick 1 $\rightarrow$ Location 3 - store 2 & pick 2 $\rightarrow$ depot], it can be rearranged as : [depot $\rightarrow$ Location 3 - store 2 & pick 2 $\rightarrow$ Location 1 - store 2 & pick 1 $\rightarrow$ Location 2 - pick 1 $\rightarrow$ depot] or [depot $\rightarrow$ Location 1 - store 2 & pick 1 $\rightarrow$ Location 3 - store 2 & pick 2 $\rightarrow$ Location 2 - pick 1 $\rightarrow$ depot]. Other combinations are infeasible due to vehicle capacity restrictions. Depending on the distance, unit load handling in each route, and location rearrangement total time can vary. The trip with the least time is selected. Total time for each selected trip is the total route time. Above steps are replicated with a different location visiting pattern for a given number of iterations before selecting the best route. Detailed pseudo code for the Random location visit heuristic is given in Algorithm 5 in Appendix A.
4.4.2 Closest location visit route construction heuristic

In the Random location visit heuristic, the next location to visit is selected randomly. In the Closest location visit heuristic, the next location to visit is selected based on the distance. This heuristic constructs routes by visiting locations that are closest to the depot in the first step of each trip and if the vehicle capacity allows visiting locations closer to the current location in the same trip. However, the starting location in the first trip is selected randomly despite starting location in each subsequent trip is selected as the closest location to the depot with a store or pick demand.

Alternatively, the starting location of the first trip can be selected as the closest location to the depot. In that case, the starting location would always be the same for a given problem instance. Therefore, all constructed routes would always be the same for all replications. This is avoided by choosing a random location as the starting point in the first trip instead of the location with shortest distance. The steps of the Closest location visit heuristic are as follows:

**Step 1.** Select a random starting location \( i_0 \).

**Step 2.** Visit location \( i_0 \). Store and/or deliver maximum possible demand.

**Step 3.** If the store and pick demand of the location \( i_0 \) is fulfilled, calculate Excess vehicle capacity; If it is \( > 0 \) go to Step 7; If it is \( = 0 \) return to the depot and go to Step 4. Otherwise return to the depot and go to Step 2.

**Step 4.** Select the location closest to the depot with unfulfilled store or pick demand (\( i \)). If demand of all locations is fulfilled go to Step 9.

**Step 5.** Visit location \( i \). Store and/or deliver maximum possible demand.

**Step 6.** If the store and pick demand of the location \( i \) is fulfilled, calculate Excess vehicle capacity; If it is \( > 0 \) go to Step 7; If it is \( = 0 \) return to the depot and go to Step 4. Otherwise return to the depot and go to Step 5.

**Step 7.** Find the closest location to the current location with unfulfilled store or pick demand (\( j \)).

**Step 8.** If the store or pick demand of location \( j \) can be fulfilled with Excess vehicle capacity or exceeds vehicle capacity, set \( i \leftarrow j \) and go to Step 5. Otherwise return to depot and go to Step 4.
Step 9. For each constructed route, rearrange visiting order by pairwise exchanging destinations.

Step 10. Check feasibility of and calculate operating time for each rearranged route. Choose the set of routes with least time ($BestRoute$).

Step 11. Calculate Total time $T$ for the routes.

Step 12. If Total time $T$ is less than $T_{\text{min}}$ update $BestRoute$.

Step 13. If the stopping criterion (number of iterations) is satisfied, output $BestRoute$ and $T_{\text{min}}$; otherwise return to Step 1.

The Closest location visit heuristic starts with selecting a random starting location $i_0$. Either the demand of location $i_0$ or maximum number of unit loads the vehicle can carry is stored and similarly picked the demand of the location; if the vehicle capacity is full after picking the demand of location $i_0$, trip is terminated by returning to the depot. The location $i_0$ is visited until its store and pick demand is fulfilled. Once the demand of the location $i_0$ is fulfilled, if the vehicle does not have excess capacity, trip is terminated and returned to the depot. In case the vehicle has excess capacity next location to visit ($j$) is selected. The location $j$ is the closest location to the location $i_0$ with an unfulfilled store or pick demand. The location $j$ is visited only if the excess capacity can fulfill the store or pick demand of location $j$ or if the store or pick demand of location $j$ requires multiple visits. Otherwise, trip is terminated and the vehicle is returned to the depot.

After completing the demand of the location $i_0$ and returning to the depot, the next location to visit ($i$) is selected as the location closest to the depot with an unfulfilled store or pick demand. As before, the location $i$ is visited until the store and pick demand is fulfilled. Once the demand of location $i$ is fulfilled, depending on the excess vehicle capacity, the vehicle either returns to the depot or visits the next location to visit ($j$) which is the closest location to the current location $i$ with an unfulfilled store or pick demand. Above steps are repeated until demand of each location is fulfilled. Once the routes are constructed, they are rearranged similar to the Random location visit heuristic and the best solution is selected. These steps are replicated with a different starting location for a given number of iterations before selecting the best route. Algorithm 6 in Appendix A shows detailed pseudo code for the Closest location visit heuristic.
4.4.3 Shortest-time location visit route construction heuristic

As in the Closest location visit heuristic, the starting location of the first trip is selected randomly in the Shortest-time location visit heuristic as well. However, instead of visiting locations in close vicinity (based on distance), total travel and unit load handling time is considered when next location to visit is selected. This heuristic builds routes based on adding locations that can be reached and serviced in the minimum amount of time. The steps of the Shortest-time location visit heuristic is as follows:

**Step 1.** Select a random starting location $i_0$.

**Step 2.** Visit location $i_0$. Store and/or deliver maximum possible demand.

**Step 3.** If store and pick demand of the location $i_0$ is fulfilled, calculate Excess vehicle capacity; If it is $> 0$ go to Step 7; If it is $= 0$ return to the depot and go to Step 4. Otherwise return to the depot and go to Step 2.

**Step 4.** Select the location with shortest route time from the depot with unfulfilled store or pick demand ($i$). If demand of all locations is fulfilled go to Step 9.

**Step 5.** Visit location $i$. Store and/or deliver maximum possible demand.

**Step 6.** If store and pick demand of the location $i$ is fulfilled, calculate Excess vehicle capacity; If it is $> 0$ go to Step 7; If it is $= 0$ return to the depot and go to Step 4. Otherwise return to the depot and go to Step 5.

**Step 7.** Find the location with shortest route time from the current location with unfulfilled store or pick demand ($j$).

**Step 8.** If the store or pick demand of location $j$ can be fulfilled with Excess vehicle capacity or exceeds vehicle capacity, set $i \leftarrow j$ and go to Step 5. Otherwise return to depot and go to Step 4.

**Step 9.** For each constructed route, rearrange visiting order by pairwise exchanging destinations.

**Step 10.** Check feasibility of and calculate operating time for each rearranged route. Choose the set of routes with least time ($BestRoute$).

**Step 11.** Calculate Total time $T$ for the routes.

**Step 12.** If Total time $T$ is less than $T_{\text{min}}$ update $BestRoute$.

**Step 13.** If the stopping criterion (number of iterations) is satisfied, output $BestRoute$ and $T_{\text{min}}$; otherwise return to Step 1.
As before, the *Shortest-time location visit* heuristic starts with selecting a random starting location $i_0$. Location $i_0$ is visited to store and pick demand; if the vehicle capacity is full after picking the demand of location $i_0$, trip is terminated and the vehicle is returned to the depot. The location $i_0$ is visited until its store and pick demand is fulfilled. Once the demand of the location $i_0$ is fulfilled and if the vehicle does not have excess capacity, trip is terminated and the vehicle is returned to the depot. If the vehicle has excess capacity, the next location to visit $j$ is selected. $j$ is the location that can be reached and served fastest from location $i_0$. The location $j$ is visited only if the excess capacity can fulfill the store or pick demand of location $j$ or if the store or pick demand of location $j$ requires multiple visits. Otherwise, trip is terminated and returned to the depot.

Each subsequent trip that starts from the depot is started at the location with the minimum route time from the depot. That is, the next location to visit $(i)$ is selected as the location with smallest travel and unit load handle time with an unfulfilled store or pick demand. The location $i$ is visited until the store and pick demand is fulfilled. Once the demand of the location $i$ is fulfilled, depending on the excess vehicle capacity, the vehicle either returns to the depot or visits the next location to visit $(j)$ which is the location with minimum route time from the current location $i$ with an unfulfilled store or pick demand. Above steps are repeated until demand of each location is fulfilled. Once the routes are constructed, they are rearranged similar to the *Random location visit* heuristic and the best solution is selected. These steps are replicated with a different starting location for a given number of iterations before selecting the best route. Algorithm 7 in Appendix A shows detailed pseudo code for the *Shortest-time location visit* heuristic.

### 4.4.4 Longest-time location visit route construction heuristic

In the *Longest-time location visit* heuristic, each route starting from the depot uses the location that requires longest travel and handle time from the depot as the first location to visit. In other words, routes are constructed from visiting location in an out-to-in order of the service time of each location. However, the first location to visit
is selected randomly, not based on the service time. If the vehicle has excess capacity after visiting the first location in each route, the next location to visit in the same trip is selected as the location which can be serviced fastest from the current location. The steps of the *Longest-time location visit* heuristic is as follows:

**Step 1.** Select a random starting location \( i_0 \).

**Step 2.** Visit location \( i_0 \). Store and/or deliver maximum possible demand.

**Step 3.** If store and pick demand of the location \( i_0 \) is fulfilled, calculate *Excess vehicle capacity*; If it is > 0 go to Step 7; If it is = 0 return to the depot and go to Step 4. Otherwise return to the depot and go to Step 2.

**Step 4.** Select the location with shortest route time from the depot with unfulfilled store or pick demand \( (i) \). If demand of all locations is fulfilled go to Step 9.

**Step 5.** Visit location \( i \). Store and/or deliver maximum possible demand.

**Step 6.** If store and pick demand of the location \( i \) is fulfilled, calculate *Excess vehicle capacity*; If it is > 0 go to Step 7; If it is = 0 return to the depot and go to Step 4. Otherwise return to the depot and go to Step 5.

**Step 7.** Find the location with shortest route time from the current location with unfulfilled store or pick demand \( (j) \).

**Step 8.** If the store or pick demand of location \( j \) can be fulfilled with *Excess vehicle capacity* or exceeds vehicle capacity, set \( i \leftarrow j \) and go to Step 5. Otherwise return to depot and go to Step 4.

**Step 9.** For each constructed route, rearrange visiting order by pairwise exchanging destinations.

**Step 10.** Check feasibility of and calculate operating time for each rearranged route. Choose the set of routes with least time \( (BestRoute) \).

**Step 11.** Calculate Total time \( T \) for the routes.

**Step 12.** If Total time \( T \) is less than \( T_{\text{min}} \) update \( BestRoute \).

**Step 13.** If the stopping criterion (number of iterations) is satisfied, output \( BestRoute \) and \( T_{\text{min}} \); otherwise return to Step 1.

In the *Longest-time location visit* heuristic, first location to visit \( (i_0) \) is randomly selected. Location \( i_0 \) is visited to store and pick demand; if the vehicle capacity is full after picking the demand of location \( i_0 \), trip is terminated and the vehicle is returned to the depot. The location \( i_0 \) is visited until its store and pick demand is fulfilled. Once the
demand of the location $i_0$ is fulfilled and if the vehicle does not have excess capacity, trip is terminated and the vehicle is returned to the depot. If the vehicle has excess capacity, the next location to visit $j$ is selected. $j$ is the location that can be reached and served fastest from location $i_0$. The location $j$ is visited only if the excess capacity can fulfill the store or pick demand of location $j$ or if the store or pick demand of location $j$ requires multiple visits. Otherwise, trip is terminated and returned to the depot.

The next first location to visit ($i$) in each trip is selected as the location with the longest travel and unit load handle time. The location $i$ is visited until the store and pick demand is fulfilled. Once the demand of the location $i$ is fulfilled, if the vehicle has no excess vehicle capacity it returns to the depot and trip is terminated. In case the vehicle has excess capacity, the next location to visit ($j$) is selected as the location with minimum route time from the current location $i$ with an unfulfilled store or pick demand. The location $j$ is visited only if the excess capacity can fulfill the store or pick demand of location $j$ or if the store or pick demand of location $j$ requires multiple visits. Otherwise, trip is terminated and returned to the depot.

Above steps are repeated until demand of each location is fulfilled. Once the routes are constructed, they are rearranged similar to the Random location visit heuristic and the best solution is selected. These steps are replicated with a different starting location for a given number of iterations before selecting the best route. Algorithm 8 in Appendix A shows detailed pseudo code for the Longest-time location visit heuristic.
4.5 Numerical Study

To study performance of the above heuristics, several numerical experiments were conducted. The exact optimization model was developed using IBM ILOG CPLEX Optimization Studio Version 12.3 and run on a personal computer with an Intel Xeon E5645 CPU running at 2.4 GHz, 4 GB of RAM, and running Windows 7. Heuristic methods were developed using Microsoft Visual Basic for Applications (VBA) with an Excel front end and run on a personal computer with an Intel Core i7-3720QM processor running at 2.5 GHz, 8 GB of RAM, and running Windows 10.

When comparing performance measures in this chapter, the percentage difference is computed as follows:

\[
\text{Difference in \%} = \frac{\text{Maximum value} - \text{Minimum value}}{\text{Minimum value}} \times 100
\]

The percentage decrease/increase of a value is computed as follows:

\[
\text{Increase/ decrease in \%} = \left| \frac{\text{Original value} - \text{Changed value}}{\text{Original value}} \right| \times 100
\]

4.5.1 Comparison of heuristic methods

In order to evaluate the performance of the proposed heuristics, 10 test problems with 20 locations are studied. Chapter 3 focuses on multi-command operations when unit load handling vehicle can carry maximum of two unit loads. In this study, vehicle capacity can be more than 2 unit loads. Therefore, for this numerical study the unit load handling vehicle capacity (Q) is assumed to be 4 unit loads. Although the heuristics allow vehicle capacity to be more than 4, this seems to be the maximum seen in practice.

Although maximum number of unit loads per location can be any positive number, it is set to 7 in this numerical study to make unit load carrying vehicle visit a location more than once to fulfill store or pick demand. Furthermore, a location can have store only, pick only or pick & store demand. To reflect this, the pick and store demands are
generated as uniformly distributed values between [0,7] for each location. This enables the unit load handling vehicle to visit some locations only once and some locations more than once, even in a simple dual-command heuristic.

The unit load handling vehicle traverses at a rate \( v \) of 300 feet per minute, well within the forklift speed limits in a warehouse ([14]). The time required to pick a single unit load from a location \( t_p \) is 0.3 minutes and the time required to store a single unit load at a location \( t_s \) is 0.3 minutes ([102], [18]).

The MILP model is solved using CPLEX 12.6 solver and the best integer feasible solution given by CPLEX after 900 seconds is recorded. It should be noted that CPLEX could not solve a 20 location problem instance to an optimal solution after running for 3 hours. Figure 4.1 shows the variation of objective function value over time in the CPLEX engine. Run time of 900 seconds was selected because the feasible solutions did not improve considerably after few minutes of run time. All heuristics were run for 50 iterations and the solution with the best objective value was selected. A sample test problem is given in Appendix B Table 4 and a sample set of routes generated by the Shortest-time location visit heuristic in one iteration is given in Appendix B Table 5.

Figure 4.1: Objective function value variation with time in the MILP method for a 20 location problem

Table 4.1 shows the traveling and unit load handling times of solutions found by
Table 4.1: Travel and handling times for 20 location problems from different solution methods

<table>
<thead>
<tr>
<th>Instance</th>
<th>Unit loads</th>
<th>Store</th>
<th>Pick</th>
<th>Exact Travel</th>
<th>Exact Handle</th>
<th>Random Travel</th>
<th>Random Handle</th>
<th>Closest Travel</th>
<th>Closest Handle</th>
<th>Shortest Travel</th>
<th>Shortest Handle</th>
<th>Longest Travel</th>
<th>Longest Handle</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>68</td>
<td>70</td>
<td></td>
<td>23.58</td>
<td>72.79</td>
<td>31.73</td>
<td>73.63</td>
<td>27.02</td>
<td>73.93</td>
<td>27.37</td>
<td>73.83</td>
<td>25.63</td>
<td>73.76</td>
</tr>
<tr>
<td>2</td>
<td>73</td>
<td>79</td>
<td></td>
<td>29.84</td>
<td>80.43</td>
<td>36.13</td>
<td>81.21</td>
<td>33.27</td>
<td>81.36</td>
<td>30.65</td>
<td>81.59</td>
<td>29.87</td>
<td>81.17</td>
</tr>
<tr>
<td>3</td>
<td>61</td>
<td>70</td>
<td></td>
<td>26.71</td>
<td>69.53</td>
<td>33.43</td>
<td>69.94</td>
<td>28.33</td>
<td>69.92</td>
<td>29.08</td>
<td>70.47</td>
<td>28.75</td>
<td>69.93</td>
</tr>
<tr>
<td>4</td>
<td>64</td>
<td>77</td>
<td></td>
<td>24.31</td>
<td>73.71</td>
<td>32.72</td>
<td>75.70</td>
<td>29.75</td>
<td>75.54</td>
<td>28.60</td>
<td>75.74</td>
<td>28.77</td>
<td>75.46</td>
</tr>
<tr>
<td>5</td>
<td>70</td>
<td>78</td>
<td></td>
<td>29.73</td>
<td>77.55</td>
<td>37.03</td>
<td>78.77</td>
<td>30.58</td>
<td>78.94</td>
<td>32.55</td>
<td>78.99</td>
<td>30.03</td>
<td>78.72</td>
</tr>
<tr>
<td>6</td>
<td>77</td>
<td>67</td>
<td></td>
<td>33.93</td>
<td>75.94</td>
<td>44.20</td>
<td>76.82</td>
<td>37.48</td>
<td>76.86</td>
<td>34.18</td>
<td>76.77</td>
<td>35.00</td>
<td>76.63</td>
</tr>
<tr>
<td>7</td>
<td>75</td>
<td>75</td>
<td></td>
<td>29.74</td>
<td>78.50</td>
<td>38.38</td>
<td>80.03</td>
<td>36.70</td>
<td>80.18</td>
<td>37.65</td>
<td>80.36</td>
<td>31.28</td>
<td>79.77</td>
</tr>
<tr>
<td>8</td>
<td>74</td>
<td>67</td>
<td></td>
<td>35.86</td>
<td>74.03</td>
<td>45.53</td>
<td>75.33</td>
<td>40.25</td>
<td>75.61</td>
<td>37.08</td>
<td>75.66</td>
<td>36.87</td>
<td>75.27</td>
</tr>
<tr>
<td>9</td>
<td>74</td>
<td>72</td>
<td></td>
<td>29.32</td>
<td>76.29</td>
<td>46.70</td>
<td>77.75</td>
<td>30.92</td>
<td>77.79</td>
<td>31.97</td>
<td>77.94</td>
<td>30.65</td>
<td>77.54</td>
</tr>
<tr>
<td>10</td>
<td>62</td>
<td>77</td>
<td></td>
<td>27.09</td>
<td>73.09</td>
<td>34.45</td>
<td>74.56</td>
<td>28.85</td>
<td>74.29</td>
<td>31.88</td>
<td>74.17</td>
<td>29.25</td>
<td>74.13</td>
</tr>
</tbody>
</table>

The MILP method (after 900 seconds) and the four heuristics for each test problem. All travel and handle times are in minutes. The data in Table 4.1 and Table 4.2 shows that the MILP (after 900s run) provides the lowest individual travel and handle times for each problem instance even though it is not optimal. The Closest location visit, the Shortest-time location visit and the Longest-time location visit have 4.3%, 4.3%, and 2.5% higher average total time (averaged over all instances) than that of the MILP method, respectively. The Random location visit heuristic, which is a simple route construction method, gives the highest total route time for each problem instance averaging 9.8% more than that of the MILP method.

Travel times differ the most compared to the solutions found by the MILP. For instance, the Random location visit heuristic routes have 31% more average travel time compared to the average travel times by the MILP and the Longest-time location visit heuristic’s travel times are 5.5% higher than that by the MILP. The Random location visit heuristic constructs routes with highest travel times for each problem instance. Routes by the Fastest-time location visit heuristic provide shortest travel times for most of the test problems, among results from heuristics. Routes by the Longest-time location visit heuristic also have lesser travel time than the Closest location visit heuristic and the Random location visit heuristics in general. Overall, it is notable that heuristics that select next location based on total route time, i.e. the Shortest-time location visit and the Longest-time location visit heuristics, produce routes with shorter travel time.
When the unit load handling vehicle has excess capacity after fulfilling the store & pick demand of a location \( i \), the next location to visit \( (j) \) is selected. If the demand of location \( j \) can be fulfilled in a single visit from the depot, i.e. when the remaining store & pick demand of location is less than the vehicle capacity \( Q \), inter-location travel can result in longer travel times and longer total route times. Thus, when a location can be served by a single visit from the depot or if inter-location trip cannot fulfill store or pick demand of location \( j \), inter-location visits are avoided even when the vehicle has excess capacity.

In the Random location visit heuristic next location \( j \) is selected randomly and in Closest location visit heuristic it is the closest to the current location \( i \). In both cases, number of unit loads to be stored or picked at location \( j \) is not considered while selecting \( j \). On the other hand, in the Shortest-time location visit and the Longest-time location visit heuristics, next location to visit \( j \) is selected as the location with minimum total travel and handling time from location \( i \). Therefore both the number of unit loads to be handled and distance are considered when selecting the next location. As a result, next location, \( j \), better because it has been selected based on both travel and handling aspects. When a vehicle has excess capacity and the next location to visit is selected randomly or based on the distance, that location might not be visited in the same trip to avoid unnecessary inter-location visits. However, when next location is selected considering the travel and handle time, it is likely that the location \( j \) can be visited in the same trip, thus shortening the total travel time. As a result, the Shortest-time location visit and the Longest-time location visit heuristics are able to construct routes with the shorter travel times.

Routes constructed by the heuristics have higher handling times compared to that from the exact method for each problem instance. However, they are at most 2.7% higher than the handling times found by the exact method. Furthermore, variation between handling times from all heuristic is less than 1% for each test problem. The Longest-time location visit heuristic constructs routes with the least handling time for most of the test
problems. In instance 1 the *Random location visit* heuristic and in instance 3 the *Closest location visit* heuristic provide routes with least handling time. They are the two instance with the smallest number of unit loads to handle among the 10 test problems. That leaves less options to combine routes compared to other instances and as a result the *Shortest-time location visit* and the *Longest-time location visit* heuristics have less chance to improve the handling.

When routes are combined, more unit loads are handled in a single trip compared to non-combined routes. Although storing or picking more unit loads in a trip saves handling time than handling them individually, the time savings over handling more unit loads in a trip is small fraction of the total time. For example, consider a problem with 2 locations. The distances from depot to location A, depot to location B and location A to location B are 180, 295 and 195 feet, respectively. Location A has a store & pick demand of 3 and 2, respectively, and location B has a store & pick demand of 1 and 2 unit loads, respectively. To fulfill the demand by a dual-command heuristic when the vehicle capacity is 4 unit loads, two trips are required (trip 1: depot → location A - store 3 and pick 2 → return to depot. trip 2: depot → location B - store 1 and pick 2 → return to depot) with 4.45 minutes of total handling time (single unit load handling time is 0.3 minutes and multiple handling factor is 0.9). If a multi-command heuristic is used, demand of location A and B can be fulfilled in a single trip (depot → location A - store 3 and pick 1 → location B - store 1 and pick 2 → return to depot) and it requires 4.32 minutes of handling time including the unit load handling at the depot. Time savings from combining handling operations alone is mere 0.13 minutes. On the other hand, travel time savings from combining trips is 0.93 minutes. Therefore, these heuristics put more emphasis on total time savings than reducing handling time alone and thus routes that might have shorter handling times would be sacrificed for the sake of travel time savings. As a result, handling times between the compared heuristics has less variation. It is also important to note that large storage facilities have thousands of trips in a day and therefore small improvement in each trip can lead to large savings in total.
Total route time is more relevant in real-world situations than individual travel and handle times. Table 4.2 shows total route times for above heuristics for each 20 location problem instance. Figure 4.2 shows the same information.

<table>
<thead>
<tr>
<th>Instance</th>
<th>Exact</th>
<th>Random</th>
<th>Closest</th>
<th>Shortest</th>
<th>Longest</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>96.37</td>
<td>105.37</td>
<td>100.65</td>
<td>101.30</td>
<td>99.40</td>
</tr>
<tr>
<td>2</td>
<td>110.27</td>
<td>117.35</td>
<td>114.63</td>
<td>112.24</td>
<td>111.03</td>
</tr>
<tr>
<td>3</td>
<td>96.24</td>
<td>103.37</td>
<td>98.25</td>
<td>99.55</td>
<td>98.68</td>
</tr>
<tr>
<td>4</td>
<td>98.02</td>
<td>108.42</td>
<td>105.29</td>
<td>104.34</td>
<td>104.22</td>
</tr>
<tr>
<td>5</td>
<td>107.28</td>
<td>115.81</td>
<td>109.52</td>
<td>111.54</td>
<td>108.76</td>
</tr>
<tr>
<td>6</td>
<td>109.87</td>
<td>121.02</td>
<td>114.34</td>
<td>110.95</td>
<td>111.63</td>
</tr>
<tr>
<td>7</td>
<td>108.24</td>
<td>118.41</td>
<td>116.88</td>
<td>118.01</td>
<td>111.05</td>
</tr>
<tr>
<td>8</td>
<td>109.89</td>
<td>120.86</td>
<td>115.86</td>
<td>112.75</td>
<td>112.14</td>
</tr>
<tr>
<td>9</td>
<td>105.61</td>
<td>124.45</td>
<td>108.71</td>
<td>109.91</td>
<td>108.17</td>
</tr>
<tr>
<td>10</td>
<td>100.18</td>
<td>109.01</td>
<td>103.14</td>
<td>106.06</td>
<td>103.38</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>104.20</strong></td>
<td><strong>114.40</strong></td>
<td><strong>108.73</strong></td>
<td><strong>108.66</strong></td>
<td><strong>106.85</strong></td>
</tr>
</tbody>
</table>

Table 4.2: Total route time for 20 location problems from different solution methods

It is evident from the data in Table 4.2 that *Longest-time location visit* heuristic constructs routes with minimum total time for most test problems. The *Shortest-time location visit* heuristic provides routes with minimum total time in once instance and the *Closest location visit* heuristic in two instances. However, in all three cases, the *Longest-
The *Shortest-time location visit* heuristic provides routes with smallest total time due to following reason. Visiting the closest location to construct routes when the vehicle has excess capacity might oversee better routes which can reduce total time. For example, consider a vehicle has excess pick capacity of 2 unit loads. The closest location to the current location, say location A, has an unfulfilled pick demand of 1 unit load and another location B has an unfulfilled pick demand of 2 unit loads. When next location is decided based on the distance, location A is visited after current location and returned to the depot. In this case, only 3 unit loads are picked which is less than the full capacity of vehicle. However, if operation time is considered to find the next location, location B might be visited which uses the full capacity and reduce total time. Therefore considering the store & pick operation time in addition to the distance from current location to construct routes creates more opportunities to serve in the same trip, which would reduce overall route time. As a result heuristics that consider operation time, i.e. the *Longest-time location visit* and the *Shortest-time location visit*, have the smallest total route times for most of the test problems.

It is difficult to compare the performance of each proposed heuristic by considering objective values of individual problem instances because characteristics of each problem instance affects the performance of the heuristics. Therefore comparing average route time across all the test problems for a heuristic is a better way to compare the heuristics.

Table 4.3 compares the best individual travel and handle times and total route time from proposed heuristics with the times given by the MILP solution. (Since the data shows the minimum travel and handle times from all heuristics for each instance, adding minimum travel and handle time for a given problem instance does not give minimum total time. For example, the minimum handling time for problem instance 1 (73.63 min) is provided by the *Random location visit* heuristic and the minimum travel time (25.63 min) is provided by the *Fastest-time location visit* heuristic. However, minimum total route
<table>
<thead>
<tr>
<th>Instance</th>
<th>Exact Travel Time (min)</th>
<th>Min. Heuristic Travel Time (min)</th>
<th>Gap (%)</th>
<th>Exact Handling Time (min)</th>
<th>Min. Heuristic Handling Time (min)</th>
<th>Gap (%)</th>
<th>Exact Total Time (min)</th>
<th>Min. Heuristic Total Time (min)</th>
<th>Gap (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>23.58</td>
<td>25.63</td>
<td>8.72</td>
<td>72.79</td>
<td>73.63</td>
<td>1.16</td>
<td>96.37</td>
<td>99.41</td>
<td>3.15</td>
</tr>
<tr>
<td>2</td>
<td>29.84</td>
<td>29.87</td>
<td>0.08</td>
<td>80.43</td>
<td>81.17</td>
<td>0.92</td>
<td>110.27</td>
<td>111.03</td>
<td>0.69</td>
</tr>
<tr>
<td>3</td>
<td>26.71</td>
<td>28.33</td>
<td>6.09</td>
<td>69.53</td>
<td>69.92</td>
<td>0.56</td>
<td>96.24</td>
<td>98.25</td>
<td>2.09</td>
</tr>
<tr>
<td>4</td>
<td>24.31</td>
<td>28.6</td>
<td>17.65</td>
<td>73.71</td>
<td>75.46</td>
<td>2.37</td>
<td>98.02</td>
<td>104.22</td>
<td>6.33</td>
</tr>
<tr>
<td>5</td>
<td>29.73</td>
<td>30.03</td>
<td>1.02</td>
<td>77.55</td>
<td>78.72</td>
<td>1.51</td>
<td>107.28</td>
<td>108.76</td>
<td>1.48</td>
</tr>
<tr>
<td>6</td>
<td>33.03</td>
<td>34.18</td>
<td>0.74</td>
<td>76.59</td>
<td>76.63</td>
<td>0.91</td>
<td>109.87</td>
<td>110.95</td>
<td>0.98</td>
</tr>
<tr>
<td>7</td>
<td>29.74</td>
<td>31.28</td>
<td>5.20</td>
<td>78.5</td>
<td>79.77</td>
<td>1.62</td>
<td>108.24</td>
<td>111.05</td>
<td>2.60</td>
</tr>
<tr>
<td>8</td>
<td>35.86</td>
<td>36.87</td>
<td>2.82</td>
<td>74.03</td>
<td>75.27</td>
<td>1.68</td>
<td>109.89</td>
<td>112.14</td>
<td>2.05</td>
</tr>
<tr>
<td>9</td>
<td>29.32</td>
<td>30.63</td>
<td>4.49</td>
<td>76.29</td>
<td>77.54</td>
<td>1.63</td>
<td>105.61</td>
<td>108.17</td>
<td>2.43</td>
</tr>
<tr>
<td>10</td>
<td>27.09</td>
<td>28.85</td>
<td>6.51</td>
<td>73.09</td>
<td>74.13</td>
<td>1.42</td>
<td>100.18</td>
<td>103.14</td>
<td>2.95</td>
</tr>
<tr>
<td>Average</td>
<td>29.01</td>
<td>30.43</td>
<td>5.33</td>
<td>75.19</td>
<td>76.22</td>
<td>1.38</td>
<td>104.2</td>
<td>106.71</td>
<td>2.46</td>
</tr>
</tbody>
</table>

Table 4.3: Comparison of heuristic results with exact method results in 20 location problems

The time provided by the Fastest-time location visit heuristic (99.40 min) is not equivalent to the summation of minimum handle and travel times.

Overall, the minimum handling time from heuristic methods have less than 2% gap compared to handling times from exact solution although highest gap can be as high as 2.4%. The minimum travel time from heuristic methods show approximately a 5% gap compared to travel times from exact solution and as high as 17.7% for some problem instances leaving room for improvement. On average, the best total route time from best heuristic methods is within 2.5% of the total time from exact method. The minimum total route time from the Longest-time location visit is 2.54% higher compared to the total route time from exact method (Table 4.2). Total route time from the Shortest-time location visit, the Closest location visit, and the Random location visit heuristics are, respectively, 4.29%, 4.35% and 9.8% are higher compared to the exact method. Since the heuristics construct routes that are within 10% of the objective value from exact method, we assert that the proposed heuristics are able to solve the problems with reasonable accuracy compared to the MILP method.

Total route times of each solving method varies for each problem instance as evident from the data in Table 4.2. This can be due to several reasons, including the characteristics of the test problem. In Table 4.4, test problems are sorted in ascending order based on the number of unit loads, average distance between locations, total route time by exact method, total route time by the Random location visit heuristic, total route time by the
"Closet location visit" heuristic, total route time by the "Shortest-time location visit" heuristic, and total route time by the "Longest-time location visit" heuristic, respectively.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>131.00</td>
<td>3 147.53</td>
<td>3 96.24</td>
<td>3 103.37</td>
<td>3 98.25</td>
<td>3 99.55</td>
</tr>
<tr>
<td>1</td>
<td>138.00</td>
<td>1 152.47</td>
<td>1 96.37</td>
<td>1 105.37</td>
<td>1 100.95</td>
<td>1 101.20</td>
</tr>
<tr>
<td>10</td>
<td>139.00</td>
<td>7 162.34</td>
<td>4 98.40</td>
<td>4 108.42</td>
<td>10 103.14</td>
<td>4 104.34</td>
</tr>
<tr>
<td>4</td>
<td>141.00</td>
<td>10 169.52</td>
<td>10 100.18</td>
<td>10 109.01</td>
<td>4 105.29</td>
<td>10 106.06</td>
</tr>
<tr>
<td>8</td>
<td>141.00</td>
<td>2 171.38</td>
<td>9 105.61</td>
<td>5 115.81</td>
<td>9 108.71</td>
<td>9 109.91</td>
</tr>
<tr>
<td>6</td>
<td>144.00</td>
<td>5 171.45</td>
<td>5 107.28</td>
<td>2 117.35</td>
<td>5 109.52</td>
<td>6 110.95</td>
</tr>
<tr>
<td>9</td>
<td>146.00</td>
<td>3 175.60</td>
<td>7 108.24</td>
<td>7 118.41</td>
<td>6 114.34</td>
<td>5 111.54</td>
</tr>
<tr>
<td>5</td>
<td>148.00</td>
<td>6 179.02</td>
<td>6 109.87</td>
<td>8 120.86</td>
<td>2 114.63</td>
<td>2 112.24</td>
</tr>
<tr>
<td>7</td>
<td>150.00</td>
<td>8 179.30</td>
<td>8 109.89</td>
<td>6 121.02</td>
<td>8 115.86</td>
<td>8 112.75</td>
</tr>
<tr>
<td>2</td>
<td>152.00</td>
<td>9 186.12</td>
<td>2 110.27</td>
<td>9 124.45</td>
<td>7 116.88</td>
<td>7 118.01</td>
</tr>
</tbody>
</table>

Table 4.4: Comparison of total route time with model characteristics in 20 location problems

The smallest total route time obtained from all solution methods is for instance number 3. This instance has the least number of unit loads to be stored and picked. Similarly, the second smallest total route time by all solution methods is for instance number 1 which handles second least number of unit loads; In addition, test problem no. 1 has the second least average distance between locations.

These observations suggest that characteristics of the test problems, such as average distance between locations and total number of unit load stored and picked affects the objective value. The test problems with least number of unit loads results in minimum objective values. However, when number of unit loads to be stored and picked is increased, distance between locations mostly decide the objective value order. Differences in objective value order in each solution method is due to different methods of route construction. Based on the other parameters such as vehicle speed and store and pick times, visiting order by operation time based methods on these would be different from that by distance based methods.

Figure 4.3 shows box and whisker plots of objective function values from all solution methods for the test problems. Maximum, third and first quartiles, median and minimum values of each data set is shown in the figure.

From Figure 4.3 it is clear that the "Longest-time location visit" heuristic provides
solutions with the smallest dispersion. Although the exact method provides overall smaller objective values, the Longest-time location visit heuristic has a smaller spread compared to all other heuristic methods. The Shortest-time location visit heuristic provides routes with slightly higher median objective value compared to the Longest-time location visit heuristic, but it has a wider spread relative to the latter. Similarly, median objective value by the Closest location visit heuristic is slight higher than the Longest-time location visit heuristic, but the solutions are spread wider compared to the latter. The Random location visit heuristic shows the highest spread among the solution methods.

4.5.2 Pick time variation

Location visit order can vary depending on the problem characteristics such as the unit load handling times, distance between locations, vehicle capacity, vehicle speed, etc. A set of ten problem instances were selected to further study the performance of the proposed heuristics under different conditions. In each test problem 20 locations are visited. The distance matrix for each location is varied and pick and store demands are generated as uniformly distributed values between [0,7]. It should be noted that the problem set used here is the same used in Section 4.5.1.

Unit load handling time might be different for different applications. Some applications such as handling empty unit loads might need only few seconds while some, such
as a fully loaded unit load, might take more than a minute. Therefore it is important to study the variation of unit load pick and store times.

In this scenario unit load pick time \((t_p)\) is varied from 0.1 minutes to 1.0 minutes in 0.1 minute steps. Store time \((t_s)\) is 0.3 minutes; vehicle speed \((v)\) is 300 ft/min and vehicle capacity \((Q)\) is 4 unit loads. Traveling and handling time averages across the 10 test problems is shown in Figure 4.4. Data used for Figures 4.4 and 4.5 is given in Table 6 in Appendix B.

![Figure 4.4: Total route time vs. unit load pick time variation for different route construction heuristics in 20 location problems](image)

It is evident from Figure 4.4 that handling time increases when unit load pick time is increased as intuition suggests. Handling time savings from combining routes is rather small fraction to travel time savings and it can be observed in the results. For a given unit load pick time, handling time variation between heuristics is minimal. The difference between the maximum and the minimum handling time for each pick time is less than 1% of the minimum value.

Also unit load pick time does not directly affect the travel time; however, it can affect the routing decision which indirectly changes travel time. Figure 4.5 shows the travel
times from each construction heuristic for different pick times. Unlike handle time, travel time does not increase proportionally to the pick time. Travel times of routes by Random location visit and Closest location visit heuristics vary less than 1% of the minimum value throughout the pick time range. This clearly indicates that the routes constructed by those heuristics have not considered the unit load handling time for decision making.

On the other hand, routes generated using the Shortest-time location visit heuristic show a 6.4% variation and those generated by the Longest-time location visit heuristic have a variation of 11.71% between the biggest and smallest travel times. This shows now routing decisions are affected by pick time variation for the Shortest-time location and the Longest-time location heuristics. Because they construct routes considering the unit load handling time, changing unit load pick time affects visiting order.

![Graph showing travel time vs. unit load pick time for different route construction heuristics in 20 location problems](image)

**Figure 4.5**: Travel time vs. unit load pick time for different route construction heuristics in 20 location problems

For the Longest-time location heuristic, travel time increases with unit load pick time. When pick time is low, its impact on the total route time is low. Therefore, location that can save the most travel time is chosen to be visited next. When pick time is high, the next location to visit is chosen based on the store & pick time savings. As a result,
travel time savings when the pick time is high would relatively low. Travel times of the Shortest-time location heuristic reaches a minimum when pick time is 0.4 minutes.

The Longest-time location visit heuristic provides routes with shortest travel time when unit load pick time is less than 0.6 minutes and the Closest location visit heuristic provides the routes with shortest travel time when unit load pick time is 0.6 minutes or greater. When unit load pick time is high, the next location to visit when vehicle has excess capacity is most often based on the store & pick demand, not distance. As a result, the emphasis on the travel time portion of the total route time is low. Therefore, heuristics that construct routes based on the operational time would choose to visit locations that would reduce handling time because they reduce overall operation time even though they have longer travel times. On the other hand, when the unit load pick time is low, better routes can be constructed by considering both travel and handle times compared to considering travel time alone.

Total route time increases when pick time is increased because unit load handling time increases. The Shortest-time location visit heuristic constructs routes with minimum total route time when unit load pick time is 0.5 minutes or lower. The Closest location visit heuristic constructs routes with minimum total route time when unit load pick time is greater than 0.5 minutes. When the average total route time across the unit load pick time range is considered, the Shortest-time location visit provides minimum time routes.

Since handling time for a given unit load pick time varies between heuristics, the total route time mainly depends on the difference in travel time between heuristics. The heuristic that has minimum travel time for a given pick time has the minimum total route time. As a result, the Longest-time location visit heuristic becomes the best choice for route construction when unit load pick time is 0.5 minutes or less and the Closest location visit method when pick time is more than 0.5 minutes.

Overall, the time savings from the Longest-time location heuristic at low unit load pick times is greater than the time savings from the Closest location heuristic when unit load pick time is high. For example, the gap between the maximum and minimum route
times when pick time is 0.1 minutes is 10.68%. For a pick time of 1.0 minutes, the gap is 2.74%. As a result, the *Longest-time location visit* heuristic constructs routes with overall minimum total route time.

### 4.5.3 Store time variation

To study the effects of unit load store time on route generation by the proposed heuristics, unit load store times ($t_s$) are varied from 0.1 minutes to 1.0 minutes in 0.1 minute steps. Similar to the unit load pick time variation scenario, pick time ($t_p$) is set to be 0.3 minutes; vehicle speed ($v$) is 300 ft/min and vehicle capacity ($Q$) is 4 unit loads. Test problems used in this scenario are identical to the scenarios in Section 4.5.1 which uses the same distance matrix and store and pick demand list. Traveling and handling average times across the 10 test problems are shown in Figure 4.6. Table 7 in Appendix B shows the same data.

![Figure 4.6: Total route time vs. unit load store time for different route construction heuristics in 20 location problems](image)

Similar to the unit load pick time variation, unit load store time directly affects the handling time. Unit load handling time by increases linearly when unit load store time
is increased in all heuristics. However, for a given unit load store time, handling time variation between heuristics is minimal. Differences between maximum and minimum handling times for each store time is less than 1% of the minimum value. The lack of variation can be explained by the comparatively small time savings by combining routes. Routes which would save travel time may be constructed although they may not necessarily save handling time. Overall minimum handling time is obtained by the Longest-time location visit heuristic which suggests that constructing routes considering unit load operation time is the better option.

Travel time is not directly affected by unit load store time variation. If routing decisions are made without considering the unit load handling, i.e. storing and picking operations, travel time should not vary when unit load store time is varied. This can be observed from the results in Figure 4.6. Throughout the unit load store time range, the difference between the maximum and minimum travel times of routes from the Random location visit heuristic is 0.51% of the minimum value. The same difference for routes by Closest location visit heuristic is 0.31% of the minimum value.

The difference between the maximum and minimum travel times of routes constructed by the Shortest-time location visit and the Longest-time location visit heuristics are 10.53% and 4.41% of their respective minimum times, respectively. This shows that handling time effects the decisions of operation-time-based route construction heuristics. When heuristics consider unit load handling operations to construct routes, the order in which locations are visited can change depending on the unit load handling parameters.

The travel time of routes generated by the Shortest-time location visit heuristic gradually increase with unit load store time. But the travel time of routes by Longest-time location visit heuristic does not show this gradual increase. Also, the Longest-time location visit heuristic produces the minimum travel time routes for all unit load store times except the $t_s = 0.1$ minutes case. This implies that combining routes when the vehicle has excess capacity is better if routes are are started with longest-operation-time locations compared to shortest-operation-time locations.
Adding travel time and handle time shows that the *Longest-time location visit* heuristic constructs minimum total time routes when unit load store time is 0.2 minutes or higher. When the unit load store time is 0.1 minutes, the *Shortest-time location visit* heuristic provides routes with smaller total route times. Overall, solutions from the *Longest-time location visit* and the *Shortest-time location visit* heuristics do not differ much from each other. The total time gap between each heuristics is less than 3% throughout the unit load store time range. In addition, both operation-time-based location visit heuristics outperform distance-based and random location visit heuristics.

In summary, the unit load store time variation directly affects unit load handling time. But difference between each heuristic for a given store time is caused by the difference of routing methods. Operation-time-based location visit heuristics, namely the *Shortest-time location visit* and the *Longest-time location visit* heuristics construct minimum total time routes.

### 4.5.4 Store and pick time variation

In some situations, the time to store and pick a unit load would be the same. In this experiment, the unit load store time and pick times are varied together from 0.1 minutes to 1.0 minutes in 0.1 minute steps. It is assumed that the store and pick times for a single unit load are equal. The same set of 10 problems with 20 locations is used. The vehicle speed is kept constant at 300 ft/min and the vehicle capacity is assumed to be 4 unit loads. The average traveling and handling times across the 10 test problems are shown in Figure 4.7. Data used for Figure 4.7 is given in Table 8 in Appendix B.

Increasing the unit load pick and store time directly affects the unit load handling time. This is evident in Figure 4.7 where the unit load handling time increases linearly with unit load store & pick time. However, similar to individual store and pick time variations, the variation of handle time between each heuristic for a given unit load store & pick time stays below 1% of the minimum value. Combining routes has a minor effect on unit load handle time even for higher unit load store and pick times. Although
the difference is small, the Longest-time location visit heuristic provides routes with the minimum handling time for a given store & pick time throughout the range.

Travel time is not directly affected by the unit load store & pick time. Despite handling time increases with unit load store & pick time, travel time stays relatively constant throughout the store & pick time range, especially when routes are constructed without considering the unit load handling time. The Random location visit and the Closest location visit heuristics have, respectively, 0.33% and 0.17% gap between maximum and minimum travel times. This confirms that routes by the Random location visit and the Closest location visit heuristics are not affected by unit load store & pick time variation.

Travel times from Shortest-time location visit and Longest-time location visit heuristics show bigger variation throughout the store & pick time range. The gap between maximum and minimum travel times are, respectively, 3.84% and 7.82%. Also, travel time gradually increases with store & pick time for them. When store & pick time is low, handling time is a minor portion of the operation time. Therefore, selecting the next location to build a route is more flexible and provides routes with shorter time. When
store & pick time increases, handling time becomes the major portion of operation time and leaves fewer options for the next location that reduces travel time. As a result, travel time savings is low when store & pick time is increased.

The *Longest-time location visit* heuristic constructs routes with minimum travel time for a given store & pick time throughout the range while the *Shortest-time location visit* heuristic gives second best routes. Again, starting with the longest-operation-time locations leaves opportunity to visit locations with smaller-operation-times and combine routes that reduces travel time. Starting with shortest-operation-time locations leave less flexibility in combining routes and thus have slightly longer travel times.

When unit load store & pick time is low, there is more opportunity to construct routes that take advantage of shorter travel time. On the other hand when unit load store & pick time is high, travel time savings by combining routes is a small fraction of the total time. This can be observed from the data where the *Longest-time location visit* heuristic provides routes with minimum time. Also, the difference between total time by the *Longest-time location* heuristic and other methods is higher when unit load store & pick time is low. Although total route time by the *Shortest-time location visit* heuristic is not the minimum, it is still lower than total time by the *Closest location visit* and *Random location visit* heuristics for all unit load store & times.

In conclusion, constructing routes by selecting locations based on operation time proves to be a better choice than random or distance based methods.

### 4.5.5 Vehicle speed variation

Unit load handling vehicle speed depends on many factors such as vehicle’s maximum speed, warehouse physical parameters like size, and aisle width, load related parameters, safety regulations, etc. It is important to understand how the vehicle speed can affect route construction decisions.

In this scenario, vehicle travel speed ($v$) is varied from 200 ft/min to 600 ft/min in 100 ft/min steps to study the effects of speed on the performance of the heuristics. It is
assumed that the unit load store time is equal to the unit load pick time and is 0.3 minutes ($t_s = t_p = 0.3$ min). As before, vehicle capacity is assumed to be 4 unit loads. The same set of ten test problem instances that have 20 locations is used. Average traveling and handling time across the 10 test problems is shown in Figure 4.8. Table 9 in Appendix B shows the same data.

![Figure 4.8: Total route time vs. vehicle speed for different route construction heuristics in 20 location problems](image)

Obviously, vehicle speed directly affects travel time, and not unit load handling time as can be observed in Figure 4.8, where unit load handling time stays virtually constant throughout the speed range. Differences between the maximum and minimum handling times for a given heuristic throughout the vehicle speed range is less than 0.2% of the minimum value. For a given vehicle speed, handling time difference between each heuristic is less than 0.5% of the minimum time.

Also obvious from Figure 4.8 is that the vehicle speed is inversely proportional to the travel time for each heuristic. For a given vehicle speed, the Longest-time location visit heuristic constructs shortest travel time routes throughout the speed range. That is, considering unit load operation times to find locations in a route is the better option.
When vehicle speed is at the lower end, travel time is the major part of the total
time. Starting routes at locations with the highest operation time enables the heuristic to
build routes to locations with smaller operation times and reduce travel time. Therefore
travel time savings by combining routes is higher when vehicle speed is low. However,
when vehicle speed increases, total travel time decreases and travel time savings diminish.
This is evident from the data where travel time difference between the *Longest-time location visit* heuristic and other heuristics is higher when vehicle speed is low. For example, travel time gap between the *Longest-time location visit* and the *Random location visit* heuristics are 6.09% when vehicle speed is 200 ft/min and 3.49% when vehicle speed is 500 ft/min.

Total route time is inversely proportional to the vehicle travel speed, similar to
tavel time. The *Longest-time location visit* heuristic provides routes with shortest total
route time. This is as a result of having minimum travel and handle times for a given
vehicle speed. Total route time savings from the *Longest-time location visit* heuristic reduces when vehicle speed is increased, parallel with travel time savings reduction. The *Shortest-time location visit* and the *Closest location visit* heuristics construct routes with similar total time. Total time difference between each other is less than 1% of the minimum value for each vehicle speed. *Random location visit* method constructs routes with highest total route time, among compared.

In addition to the average total route time, minimum and maximum for each
scenario can be considered. Figure 4.9 shows the minimum and maximum total route
time for each heuristic along with average total route time. Data is available in Table 10
in Appendix B.

From the Figure 4.9 it can be observed that total route times for test problems
are widely spread when vehicle speed is lower. The spread narrows when vehicle speed is
increased and there is no significant drop at a certain point. The minimum and maximum
spread as a percentage of the average objective value also narrows when vehicle speed is
increased. For example, when vehicle speed is 200 ft/min, the minimum and maximum
values are 10% and 8.85% from average total time for the Closest location visit heuristic. When vehicle speed is 600 ft/min, the minimum and maximum values are 9.2% and 6.4% from the average total time, respectively. When vehicle speed is increased, required travel time decreases. As a result, the total route times becomes smaller and the minimum-maximum spread becomes narrower when vehicle speed is increased. Narrowest spread of total time for test problems is from the Longest-time location visit heuristic.

In conclusion, constructing routes with the Longest-time location visit heuristic proves to be a better choice across the range of unit load carrying vehicle speeds, especially when vehicle speed is low.

4.5.6 Vehicle capacity variation

Number of unit loads a vehicle can carry is an important parameter in routing decisions. In Chapter 3 it was assumed that a vehicle can carry maximum of 2 unit loads at a time. In this chapter it is assumed that a vehicle can carry 2 or more unit loads between two locations. To test the proposed heuristics under different vehicle capacities, unit load handling vehicle capacity ($Q$) is varied from 2 to 6 in this scenario. Same set of
10 test problem instances in which 20 locations are visited is used in this scenario as well. Single unit load store and pick time is set to 0.3 minutes and the vehicle traverse speed is set to 300 ft/min. Traveling and handling time averages across the 10 test problems is shown in Figure 4.10. Data used in Figure 4.10 is given in Table 11 in Appendix B.

![Graph showing total route time vs. vehicle capacity for route construction heuristics in 20 location problems](image)

From the results shown in Figure 4.10 it is clear that handle times stays relatively unchanged for a given vehicle capacity. The gap between maximum and minimum handling times for a given vehicle capacity is less than 0.5%. When number of unit loads, unit load store & pick time, and vehicle capacity are fixed, route constructing by visiting multiple locations has a small impact on the handling time compared to the travel time. Therefore, the variation of handling time between routes from each heuristic is small.

However, a decrease in unit load handling time can be observed when vehicle capacity is increased. Handling time savings from increasing vehicle capacity to 6 unit loads from 2 is around 9% for each heuristic. This result is expected since handling more unit loads in a single trip can save total handling time. The *Longest-time location visit* heuristic constructs routes with minimum handling time for all vehicle capacities.
When vehicle can carry more unit loads in a single trip, number of trips required to fulfill store & pick demands reduces. Therefore number of unit loads a vehicle can carry directly affects travel time. This is reflected in the results as travel time reduction when vehicle capacity is increased. Also, it should be noted that travel time is not directly proportional to the vehicle capacity. Number of trips is not halved when vehicle capacity is doubled.

The *Longest-time location visit* heuristic constructs routes with shortest travel time when vehicle capacity is 4 or less unit loads and the *Closest location visit* heuristic constructs minimum travel time routes when vehicle capacity is greater than 5 unit loads. Also, travel time by the *Shortest-time location visit* heuristic is less than travels times from the *Closest location visit* heuristic when vehicle capacity is 3 or less.

When vehicle capacity is low, it is important to select locations that save time by visiting multiple locations in the same route. If next location to visit is selected randomly or based on distance, store or pick demand of that location is not considered. As a result, the selected location might not have a demand that can be served in the same trip. On the other hand, if operation time is considered when selecting the next location, it would be easier to find a location that needs to be served in the same trip which would lead to save travel time. However when vehicle capacity is high, possibility of having excess capacity after visiting a location is high. As a result, if next location to visit can be selected without considering the store & pick operation time at the destination. Thus, chance of serving that location in the same trip will be high. On the other hand, when vehicle capacity is high, handling time becomes the major contributor to the total time. Therefore, heuristics that construct routes considering operation-time puts more emphasis on handling time which would not select the routes with smaller travel time. This explains why the *Longest-time location visit* heuristic constructs minimum travel time routes when vehicle capacity is 4 or less while the *Closest location location visit* heuristic constructs otherwise.

Overall, the total route time decreases when vehicle capacity increases for each
heuristic. Ability to carry more unit loads in a trip reduces travel time by shortening number of trips. In addition, handling time is reduced by handling more unit loads in a single trip. The Longest-time location heuristic constructs routes with minimum total route time for vehicle capacity less than or equal to 5. The Closest location visit heuristic provides the shortest total route time when vehicle capacity is 6.

The total route time gap between the Longest-time location visit heuristic and other heuristics decreases when vehicle capacity is increased. For example, when vehicle capacity is 2 unit loads, the gap is 9.19% and when vehicle capacity is 5 unit loads, the gap reduces to 6.02%. This confirms that constructing routes considering operation-times performs better when vehicle capacity is low.

![Figure 4.11: Minimum and maximum route times vs. vehicle capacity in 20 location problems](image)

When the minimum and maximum total route times for each heuristic for vehicle capacity variation is shown in Figure 4.11. (Data is available in Table 12 in Appendix B.) It is clear that the total route time is spread widely for low vehicle capacities and it narrows when vehicle capacity is increased. In addition to the minimum-maximum spread as a value, minimum-maximum spread as a percentage of the average, too, narrows when vehicle capacity is increased. For instance, the minimum and maximum total times from
the *Shortest-time location visit* heuristic for 2 unit load vehicle capacity has a gap of 11% and 7.9% from the average total time, respectively. The same values for 6 unit load vehicle capacity case reduces to 9.9% and 6.7%, respectively. When vehicle capacity is increased, number of trips required reduces and thus total route time decreases. The minimum and maximum total times, also, decreases which makes the minimum-maximum spread narrower. When vehicle capacity is high, although the *Closest location visit* heuristic provides routes with minimum total route time, it has a wider spread of solutions than other heuristics for the same vehicle capacity.

In summary, the *Longest-time location visit* heuristic becomes a better choice when number of unit loads the vehicle can carry is 5 or less while the *Closest location visit* heuristic provides better routes when vehicle capacity is 6.

### 4.5.7 Number of locations variation

Number of locations needs to be visited is a significant factor for route construction methods. In previous sections, number of locations visited is limited to 20 where at each location pick and store operations are carried out. In this section 10 problem instances each with 20, 50, 100, and 200 locations were solved using the route construction heuristics. As before, vehicle speed is set to 300 ft/min and single unit load pick and store time is set to 0.3 minutes and vehicle capacity is set to 4 unit loads. Pick and store demand is uniformly distributed between [0,7] for each location. Figure 4.12 shows traveling and handling times averaged across the 10 problem instances. The data is given in Table 13 in Appendix B.

As seen from Figure 4.12, the total handle time for a given number of locations does not vary depending on the heuristic. In fact, the gap between maximum and minimum total handling time for a given number of locations is less than 0.5%. As explained previously, handling time savings from combining routes is small compared to the travel time savings. Therefore the variation of total handling time between heuristics is small when other parameters are fixed.
The minimum handling times is provided by the Shortest-time location visit heuristic for 20 locations and 50 locations problems. the Closest location visit heuristic constructs routes with the minimum handling time for 100 locations and 200 locations problems. Visiting multiple locations in the same route is possible if vehicle has excess capacity after visiting a location and the next location has store or pick demand that can be fulfilled without increasing total time. When number of locations is low, it is important to select locations that has a demand that matches excess capacity. If next location to visit is selected randomly or only depending on the distance, the chance of finding such locations compared to finding locations depending on operation time. On the other hand when number of locations is high, the chance of the finding a location with a store or pick demand that matches the available excess capacity is high even if the locations are selected without considering the operation time. As a result, handling time of routes by the Closest location visit heuristic is lower than that by the Shortest-time location visit and the Longest-time location visit for 100 and 200 locations case.

When travel time is considered, the Longest-time location visit heuristic constructs
routes with minimum values when number of locations is 20 and 50 and the Closest location visit heuristic gives shorter time for 100 and 200 location cases. The Shortest-time location visit heuristic constructs routes with shorter travel time than the Longest-time location visit heuristic for 100 location and 200 location cases. When routes are combined based on the operation time, locations that have minimum operation time are selected as next potential visit locations. When number of locations to be visited is low, the locations are scattered in the warehouse. In that case, visiting closest location might not give the least travel time while visiting the location with lowest operation time would give minimum total time. On the other hand, when locations are densely located, chance of finding closer locations with store & pick demand is high while location with minimum operation time might not give a shorter travel time. As a result, it is effective to ignore the operation time and find locations based on the distance when the destinations are densely located.

Total route time increases with number of locations regardless of the heuristic as evident from the data. More unit loads to be stored and picked means more handling and traveling and thus total time increases for all heuristics when number of locations increase. The Random location visit heuristic construct routes with longest total time compared to all other methods. Similar to travel time, The Longest-time location visit method constructs routes with shortest total route time for 20 and 50 location cases and the Closest location visit heuristics construct routes that are shorter for 100 and 200 location problems. Both operation-time-based heuristics, i.e. the Shortest-time location and the Longest-time location provide similar routes with a less than 2% gap of total time between each other.

Table 4.13 shows the minimum and maximum total route times for each heuristic for number of locations variation. (Data is available in Table 14 in Appendix B). The minimum and maximum range increases in value with number of locations for all heuristics. For instance, the minimum is 10.5 minutes less and 8.12 minutes more compared to the average total route time from the Closest location visit heuristic for 20 locations case. The minimum and maximum values for 200 locations case are 72.77 minutes and 52.95
Figure 4.13: Minimum and maximum route times vs. number of locations

minutes below and above the average value, respectively. However, as a fraction of the average route time, minimum and maximum spread becomes narrower when number of locations are increased. For example, the minimum and maximum total route times from the Closest location visit method has a gap of 9.66% and 7.47%, respectively, from the average value for 20 location case. For 200 location case, the minimum and maximum values vary 6.91% and 5.03% from the average value. This implies that heuristics provide solutions with less variation when number of locations is increased. However, this might be due to relatively large increase of average total time compared to the minimum-maximum variation when number of visiting locations is increased. The Longest-time location visit heuristic shows narrower minimum-maximum spread when number of unit loads is 100 or less.

In summary, constructing routes by finding next location to visit based on operation time gives better routes when number of locations is low and destinations are scattered in the warehouse while constructing routes based on distance is the better option when number of locations is high.
4.5.8 Pick and store demand variation

In all problem instances considered so far, unit load store & pick demand is assumed to be uniformly distributed between [0,7] for each location. Different demand patterns for pick & store list might be significant in routing decisions. To study this, unit load store and pick demand is varied with random distribution with different values in this scenario. Store and pick demand values are uniformly distributed in [3,4], [2,5], [1,6] and [0,7] ranges. However, it is important to keep all the parameters unchanged except the variable, demand pattern in this case. Therefore randomly generated store and pick lists are manually altered such that store & pick list for each problem set has same total number of store and pick unit loads. Total number of unit loads in all replications, average number of unit loads per location and average standard deviation for each input range is given in Table 4.5. As evident from data in Table 4.5, each store & pick input range has a different standard deviation.

<table>
<thead>
<tr>
<th>Input Range</th>
<th>Total unit loads</th>
<th>Average unit loads per location</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Store</td>
<td>Pick</td>
<td>Store</td>
</tr>
<tr>
<td>3-4</td>
<td>698</td>
<td>732</td>
<td>3.49</td>
</tr>
<tr>
<td>2-5</td>
<td>698</td>
<td>732</td>
<td>3.49</td>
</tr>
<tr>
<td>1-6</td>
<td>698</td>
<td>732</td>
<td>3.49</td>
</tr>
<tr>
<td>0-7</td>
<td>698</td>
<td>732</td>
<td>3.49</td>
</tr>
</tbody>
</table>

Table 4.5: Parameters of pick and store demand lists

With above input ranges, 10 problem sets are solved with the heuristics. Vehicle speed is set to 300 ft/min with a capacity of 4 unit loads and single unit load handling time is set to 0.3 minutes. It should be noted that distance matrix for a replication remained unchanged throughout all the input ranges. In other words, distance matrix used for replication 1 of demand range [0-7] is used for replication 1 of other demand ranges as well. This enables travel and handle times to be only affected by difference in demand ranges. Average traveling and handling time across the 10 replications for each pick & store demand range is presented in Figure 4.14 and Table 15 in Appendix B.
From Figure 4.14 it can be observed that handle time stays relatively unchanged for all demand ranges. Total handling time variation is less than 1.5% of the minimum value for all input ranges. Although small, handling time increases when store & pick demand range variation increases. When demand range variation is high, some trips would carry less than vehicle capacity. When input range variation is low, most trips would carry full vehicle capacity. As a result, for a given heuristic, handling time increases when store and pick demand has a high variation. In addition, the Longest-time location visit heuristic constructs routes with minimum handling time for all input ranges.

When travel time is considered, it is notable that travel time increases when store & pick demand variation is increased for a given heuristic. Travel time difference between minimum and maximum values range from 34% to 66% of the minimum time for a given heuristic. The travel times from Random location visit heuristic increase mostly (66% increase) when store & pick demand range is changed from [3,4] to [0,7]. The travel times from Longest-time location visit heuristic shows least increase (34%).

For this scenario, it is assumed that vehicle capacity is 4 unit loads. Therefore,
when pick & store demand is either 3 or 4 unit loads per location, each location needs to be visited only once. Furthermore, in that case, excess capacity after visiting a location would be less than 2 unit loads. These factors force the vehicles to return to the depot after visiting a location and handle same number of unit loads at a location for all heuristics. As a result, handling time for all heuristics is the same for [3,4] case. In addition, all heuristics provide same routes and thus travel time is the same for all heuristics when store & pick demand is between 3 and 4. This can be observed as 0% variation of total handling time and travel time between each heuristic.

On the other hand, when the pick & store demand variation is high, store & pick demand of a location might not be able to fulfill in a single visit. In addition, some locations might be needed to visit for pick-only or store-only operations. This increases the travel time and total time. This explains the observation from the Figure 4.14 where travel time increases when demand range has a higher variation for each heuristic.

A variation of travel time between heuristics can be observed. Although handle time gap is less than 1%, travel time gap between the maximum and minimum values, on average, is 20% for [2,5],[1,6] and [0,7] demand ranges. The Longest-time location visit heuristic constructs routes with minimum travel time for all demand ranges. The reason is that the Longest-time location visit heuristic can construct routes better than other heuristics under the circumstances.

Since handling times vary less throughout the demand range, the total time varies similar to travel time. Total route time increases when store & pick demand variation increases. When individual heuristics are compared, the Longest-time location visit heuristic constructs routes with minimum total time across all demand ranges.

### 4.5.9 Computation time

In Section 4.5.1, the exact optimization method was unable to reach an optimal solution even after running for 3 hours for a 20 location problem. Finding optimal solutions for problems with more locations is practically impossible. The proposed heuristics, on
the other hand, solve the problem instances quicker and can be used to solve large problem instances. In Table 4.6, average computation time per test problem for each heuristic is presented. All computation times presented were collected while solving problems in section 4.5.7.

<table>
<thead>
<tr>
<th>Heuristic</th>
<th>20 Locations</th>
<th>50 Locations</th>
<th>100 Locations</th>
<th>200 Locations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>0.19</td>
<td>0.45</td>
<td>1.13</td>
<td>2.83</td>
</tr>
<tr>
<td>Closest</td>
<td>0.32</td>
<td>0.99</td>
<td>3.37</td>
<td>11.63</td>
</tr>
<tr>
<td>Shortest</td>
<td>0.67</td>
<td>3.29</td>
<td>14.72</td>
<td>56.54</td>
</tr>
<tr>
<td>Longest</td>
<td>0.62</td>
<td>2.99</td>
<td>14.24</td>
<td>54.60</td>
</tr>
</tbody>
</table>

Table 4.6: Average computation times for different route construction heuristic

As intuition suggests, for solving larger scale problems with more visiting location, more computation time is required for each heuristic. Furthermore, required computation time increase is not linear. It is observable from the data in Table 4.6 the Random location visit heuristic takes least time to construct routes than all other heuristics for a given set of problems while the Shortest-time location visit heuristic takes most time to construct routes. The Random location visit heuristic has the least complex algorithm for constructing routes in which locations are visited consecutively. In addition, if the vehicle has excess capacity after visiting a location, next location with a demand is visited; no parameters are calculated once a location’s demand is fulfilled. Simplicity of the algorithm results in faster route construction, although the total route times are the longest, among compared methods.

The Closest location visit heuristic uses a relatively advanced algorithm, thus needs more time to construct routes. This algorithm needs to calculate the closest locations to the depot and current location after visiting a location. This requires more computation time but it is still lower compared to operation time based methods.

Algorithm in the Shortest-time location visit heuristic is more complex compared to the Random location visit and the Closest location visit methods. For instance, the Shortest-time location visit heuristic requires to calculate total route time per location for
all the locations from the current location, compared to finding the closest location in the
*Closest location visit* heuristic to decide next location to visit. This complexity is of the
algorithm is reflected in the higher computation times in Table 4.6.

The *Longest-time location visit* heuristic also needs to calculate total route time
to each location from the depot after coming back to depot in addition to calculating the
service time to all locations from the current location after serving a location, similar to the
*Shortest-time location visit* heuristic. However as observed from the numerical examples,
the *Longest-time location visit* heuristic combines trips better than other methods. As a
result, to fulfill the store and pick demand fewer total number of trips would be required
by this method and therefore number of computations required after visiting a location
would be less than the *Shortest-time location visit* method. Therefore, computation time
of the *Longest-time location visit* heuristic is similar to that of the *Shortest-time location
visit* heuristic, it is slightly less.

In conclusion, the proposed heuristics are able to construct feasible, yet sub-optimal
routes for the problem with less computation effort. Although the routes might not
be optimal, they can be used as good practical solutions for complex multi-command
operations. Being able to solve the problems within a short amount of time enables to
update the store & pick demand lists and get new solutions in a dynamic environment.
4.6 Conclusions and Future work

In this research, warehouse operations in which vehicles can carry multiple unit loads in one route and store and pick multiple unit loads at a location was studied. A mixed-integer linear programming model was developed. The optimization model was not able to reach an optimal solution for small problems with 20 store and pick locations. The problem belongs to the NP-hard computational complexity class and thus using heuristics to find solutions is justified. Four route construction heuristics were proposed for designing routes to store and pick unit loads.

Comparing the proposed heuristics, the Longest-time location visit heuristic was proven to be the better method for constructing routes for most situations. The objective value from this heuristic shows less than 3% gap for 20 location problems compared to objective obtained from running the exact model for 900 seconds. Average computation time for problems with 200 location to visit is less than a minute for this heuristic. The Closest location visit heuristic becomes the better choice when traveling is a smaller portion of the total unit load handling operation, such as when the vehicle can carry 6 unit loads and when the locations to visit are located densely in the warehouse.

Based on different characteristics of the problem instances, one heuristic can be a better choice over another. However, in a practical application, there is no need to select one heuristic to construct routes; all heuristics can be used to construct routes and best set of routes can be selected since each heuristic can construct routes fairly fast, e.g. within a minute for a 200 location problem. The ability to construct routes within a short amount of time enables to update store and pick demand and to recalculate routes if required. This method can be used to address problems with a dynamic store and pick demand.

In terms of future research directions, applying evolutionary algorithms to the problem to find better solutions is an obvious choice for in the short term. Furthermore, the solutions by construction heuristics could be improved by local search methods. It
was assumed that only one vehicle carries out all the store & pick demand requests while constructing routes. If there are multiple vehicles, the constructed routes can be served by each vehicle, assuming that the vehicles operate independently of each other. However, if vehicles work collaboratively total route time can be reduced. Future research is required to address multiple vehicles working together to fulfill store & pick demand.

In Chapter 3, individual unit load stackability was considered. In this research, it was assumed that each unit load is stackable on any other. Adding grouping restrictions for unit loads while being carried by vehicle is a long-term future research area. All unit loads that need to be stored are located at a single location, the depot, at the beginning of the operation. Furthermore, all picked unit loads are brought back to the same location. However, there can be multiple depot locations in larger warehouses. Future research should address this possibility. In this research, it is assumed that the unit load store and pick demand has no priority or time windows for order storage and/or picking operations. Algorithms that recalculate the routes to store and/or pick priority unit loads and operate with time windows for multi-command operations are future research ideas.
Chapter 5

Intermodal Interface Operations with Time Windows and Reshipping

5.1 Introduction

Chapter 3 and Chapter 4 have illustrated why moving from distance based models to time-based models is important for making operational decisions under conditions that currently quite frequently exist in unit load warehouses like stacking pallets for transport or storing multiple unit loads at a location. In this chapter, the notion of a time based approach is used to explore some fundamental design issues in a next-generation intermodal interfaces that support a potential next-generation logistics system that features extensive collaboration. Some refer to one manifestation of this as the Physical Internet (PI).

For context, logistics and goods movement lie at the center of our quality of life, not only for the availability of consumer products but because of transportation impact on land use, energy consumption and environmental quality [9]. Logistics is a huge and important industry. The freight transportation sector (i.e., trucks, ships, and trains used to deliver freight) of the U.S. economy carries 31 million tons of freight valued at $38M every day. Truck transport that consist of private fleets, full truckload carriers (TL), and
less-than-truck load carriers (LTL) account for 73.1% of this total [5]. Assuming that the all trucks average 5.1 miles to the gallon, delivering this freight requires 30B gallons of diesel fuel which represents an annual expenditure of $120B at current prices for diesel fuel of approximately $4/gallon.

It is important to note that the logistics systems in the U.S. currently work very well most of the time with respect to reliable and timely delivery of goods; however, there is inefficiency in the system that could undermine this situation. An excellent example is truck transport. In 2007, it was estimated that trailers were 42.6% full on average as measured by the percent of maximum weight load [11]. Maybe more striking is that 25% of the miles traveled, on average, were with completely or nearly empty trailers whereas the remaining 75% were, on average, only 56.8% full [82], [83]. In economic terms, companies in the U.S spend about $30B a year shipping empty or nearly empty trailers.

With a primary fuel source for transportation and energy production being fossil fuel, it is not surprising that fossil fuel combustion accounts for 94% of the total ∼ 5,300 trillion grams (Tg) of emitted CO₂ annually in the U.S. [3]. Transportation accounts for ∼ 27% of the CO₂ emissions with the freight transportation sector alone contributing approximately 9% of the total [4]. For context, that is more than five times greater than the second-leading emitter, China with less than 100 Tg of freight-transportation-based emissions per year.

These inefficiencies impact our economy and society in many ways. Logistics prices are a major component of the cost of many goods, greenhouse gas emissions are ever-increasing, and unacceptable levels of traffic congestion are escalating. These impact each of us through the price of goods, environmental emissions, traffic jams, and an overall decrease in our quality of life. Reducing the inefficiencies of freight logistics by truck, train and intermodal platforms is part of the solution for improving economic and environmental sustainability as well as the lives of truck drivers and many Americans as well. This is the Global Logistics Sustainability Grand Challenge [89].

There are several notions on how to achieve these goals but the one most highly
developed is the PI. The PI is currently envisioned as a highly collaborative interconnected logistics system through which standard sized modular boxes flow. It is an open, global, intermodal logistics system that utilizes standard, modular, reusable containers, real-time identification, and routing through shared logistics facilities. The basic idea was patterned after certain feature of the digital internet like transmitting information under a standard TCP-IP protocol. The PI was described in Science [84] and highlights the key characteristics and advantages of PI. Implementation of PI has broad appeal on greater profits, better logistics efficiency, reduced carbon footprint, and an improved lifestyle for long haul truck drivers. For example, one possible implementation has a vast network of flexibly shared intermodal hubs in which each can be reached by at least one other hub within a one-day drive by a truck driver. Freight enters the system at any of the hubs and traverses the network to the destination one hub at a time. That is, the path from origin to destination is not predetermined; rather, the packages move from hub to hub based on capacity (e.g. to cube out a unit load or transportation container, space on trucks with destination closer to the package destination) and deadlines. To assess potential improvement, actual data from the consumer packaged goods industry was used to populate an optimization model. The results were rather impressive. If only 25% of the packages were transported using a relay strategy between intermodal hubs in the PI, fullness was increased more than 30% and the average cost per load was reduced by more than 25%. In addition, the relay idea allowed drivers to return to their home base at least every 4 days and many returned every other day with no reduction if paid miles because backhauls were full. There are, however, many serious research questions related to implementation of the PI. This research begins to explore basic relationships that are important in the design of the intermodal hubs.
5.2 Background

In an intermodal network, a shipment uses multiple modes of transportation in its journey from the origin to the destination in a seamless manner through the use of intermodal containers. Crainic and Kim (2007)[35] shows the importance of container-based intermodal transportation and discusses the strategic planning of multimodal systems based on demand, mode choice, and supply representation and assignment. They review the models for seaport container terminal operations including crane scheduling, stowage planning and sequencing, and berth scheduling. Bektas and Crainic (2008) [27] review issues and challenges in designing, planning, and operating intermodal transportation networks.

Consolidation of unit loads is a key decision made at intermodal interface. Bon-tekoning (2000) [30] describes different consolidation operations. Unit loads can be exchanged via the buffer or storage area [30]. Intermodal interface operators have to make tactical and strategic decisions on design of the terminal, type and number of equipment used, layout of the facility, etc. to meet the demand of unit load exchange. Tyan et al. (2003) [119] evaluates collaborative freight consolidation policies with a mathematical programming model. The objective is to minimize system-wide cost, which is the summation of operating cost and capacity lost cost. In this research, freight consolidation is considered as a part of internal operations of the intermodal interface.

Collaborative transportation planning for intermodal transportation which studied coordination of independent operators of different stages of intermodal transportation has been studied by Puttermann and Stadtler (2010) [104]. The suggested collaboration scheme for overseas transportation in which all parties need to exchange only non-critical data significantly reduces overall transportation costs.

The cross-docking operations have similarities with operations at intermodal interfaces. Cross-docking ([121], [31]) is used by many logistic service providers as a logistic strategy to transfer incoming shipments directly to outgoing vehicles without storing them
in between. Operational level decisions in cross-docking facilities include vehicle routing, dock door assignment, truck scheduling, and storage space management.

Pick and delivery vehicle routing associated with cross-dock environments study picking up freight from various locations and consolidating at the cross-dock and delivering to multiple locations after consolidation. Lee et al. (2006) [77] propose a distance-based tabu search algorithm to find routing schedule for pick and delivery that minimizes the transportation cost and fixed cost of the vehicles. Wen et al. [127] present tabu search algorithm for vehicle routing problem with cross-docking. In this study, it is assumed that orders are picked with a fleet of homogeneous vehicles, then consolidated at a cross-dock and immediately delivered without temporary delivered to customers by the same set of vehicles. Also, each pick and delivery location must be served by exactly one vehicle within its time window and the time horizon for whole operation must be respected.

Dock door assignment problem is focused in finding the optimal assignment dock doors to inbound and outbound trucks. In this problem, it is assumed that the number of dock doors exceed the number of trucks. Peck [100] develops a simulation model of an LTL terminal and tries to assign the trucks to dock doors in order to minimize the internal travel time of the shipments. The travel time is expressed as a function of the distance between dock doors, contents of the trucks and the type of transportation. A greedy balance algorithm is provided to solve the problem. Brown [33] studies trailer-to-door assignment problem where objective is to minimize the total travel distance. A dynamic layout, in which inbound and outbound trucks are assigned on a short-term horizon, proves to reduce the total travel distance.

When number of dock doors is less than the number of inbound and outbound trucks for a given period, inbound and outbound trucks need to be scheduled. The problem decides when the trucks should arrive/depart the cross-dock facility and which dock doors are assigned to them. Lim et al. (2006) [78] consider truck door assignment problem with time windows where number of trucks exceed the number of docks available and the objective is to minimize the total shipping distance. An integer programming
model is formulated and a tabu search and a genetic algorithm is presented to solve the problem. However, the time to transport goods between the dock doors or possibility to reship the goods is not considered. Lim et al. (2009) [85] study the same problem with internal travel time and an objective of minimizing the operational cost which is based on travel time and unfulfilled shipments. Konur and Golias (2013) [73] study a truck scheduling problem at inbound doors with unknown exact arrival times. A bi-objective optimization approach where first objective minimizes the total service cost and second objective minimizes the cost range is formulated.

Temporary storage in cross-dock environments has been studied. For instance, Vis and Roodbergen (2008) [125] study the location of the temporary storage area for incoming unit loads. Werners and Wülffing [128] study the temporary storage locations in a parcel sorting setting. The authors model the problem as a linear assignment problem where the objective is minimizing travel distance and the model determines where the freight has to be stored temporarily. Limited storage capacity with time limitations for internal operations in cross-dock environments is not addressed in the literature.

The literature on internal operations of cross-dock operations assumes that resources in the cross-dock is able to handle the goods handling demand and does not consider reshipping option. Moreover, most truck door scheduling problems assume that arrival and departure times of each truck is known. For the best of our knowledge, no research has been conducted to address limited outbound capacity and reshipping possibility.

5.3 Key Operations and Features of Intermodal Interfaces

Intermodal interfaces are a key part of the PI because they are the places where critical decisions are made and executed regarding all inbound freight. Consider inbound transportation containers. Some of these will be passed through the intermodal hub
without modification and sent to another intermodal hub. Some will be broken down and the unit loads contained therein will: 1) be combined with other unit loads into a transportation container for immediate outbound shipment, 2) stored for a brief time before being combined with other unit loads for outbound shipment, or 3) broken down into modular boxes that are either combined with other modular boxes into a unit load from further shipment or delivered locally to the final destination. The operations of the hub are high volume and can be complex involving everything from the trivial of rerouting a transportation container to handling modular boxes. The decisions must be made quickly and correctly; and the facility must be designed so that they can be executed according to the decisions. In addition evolving transport from point-to-point to distributed multi-segment intermodal transport is another key objective of physical internet [89]. Different modes of transportation (e.g. trucks, rail, air) bring transportation containers on regular basis to intermodal hubs. Each transportation container contains unit load containers and each unit load container is comprised of modular containers. Each modular container can have a different destination and possibly a delivery deadline.

At the intermodal interface, how each modular container will traverse the next leg of the route should be decided. This decision is affected by many factors including, but not limited to physical parameters, i.e. size, weight, of the modular containers, inbound schedule, available outbound transportation resources and their schedule, available resources to handle the packages at intermodal interface, cost for each decision, etc. In general, modular containers need to be disaggregated from the inbound unit loads and aggregated back in to outbound unit loads. Then outbound unit loads need to be loaded into outbound transportation containers.

Full knowledge of inbound and outbound transportation containers will be known to the intermodal interface controller before decisions must be made. For inbound freight, this will include time of arrival and complete information about the modular boxes in each inbound transportation container such as dimensions, weight, destination, and promised delivery time. This information will be known to the intermodal hub controller not
later than when the transportation container leaves the previous hub, at least half a day in virtually all cases. Some of the destinations for outbound transportation containers will likely be known well in advance because the demand on the routes, say hubs near two major cities, is reasonably constant so the routes are pre-determined for months in advance. Other outbound lanes will be opened on an as-needed basis which can only be determined as details of the inbound freight demand become clear which could be as short as half a day. The point here is that decisions in the intermodal interfaces are time sensitive and the some durations can be rather short. Hence, it is necessary to complete unloading, dis-aggregation, aggregation and loading operations at the intermodal interface within the available time frame.

Key operations of intermodal interfaces have similarities with Less-than-truckload (LTL) breakbulk operations. Both facilities disaggregate inbound bulk containers and aggregate them based on the destination. In addition, goods are not stored at the facility for a long period. However, most LTL operations are handled by a single organization and are driven by centralized information. Further, the paths between origin to destination for most packages are predetermined. There are also special situations where several Third-party logistics (3PL) operators consolidate loads at well defined, limited scale operations [64], [66], [65] that are contract negotiated. Moreover, LTL operators plan use careful central planning and control to minimize operating costs while maintaining the minimum promised level of service level [61].

In this research, it is assumed that control of the PI is decentralized and dynamic. Many operators are assumed to collaborate in the operation, and operations are driven by decentralized information that becomes available for decision making with little lead time and these assumption combine to support the notion that routes for modular containers are not predetermined. While adding complexity to the problem, this also adds tremendous flexibility for accommodating variabiilty in volume, and excess traffic conditions because it adds an opportunity to re-route modular containers to the final destination via alternate paths that are less utilized. The operational strategy of using decentralized
control and adding the possibility of routing and rerouting modular containers as they progress through the interconnected network differentiates PI operations of intermodal interfaces from LTL breakbulk operations.

5.4 Problem Description

In this research, it is assumed that tactical level operations determine the routes and schedules of transportation containers, whereas operations in intermodal interfaces are operational level. That is, it is assumed that the routes of transportation containers are predetermined but what is contained in these is dynamically determined at the intermodal interfaces. Basic operational level decisions at intermodal interface are: unloading unit loads from inbound transportation containers, disaggregating inbound unit loads into modular containers, aggregate modular containers into outbound unit loads and loading unit loads into outbound transportation containers.

In this research, we study the intermodal interfaces from an operations perspective not a design viewpoint. However, we assume that no disaggregating and aggregating modular containers is required. Emphasis of this study is on unloading inbound unit loads and assigning them into outbound transportation containers depending on the destination and capacity of each container. We assume that tactical level decisions of routes and schedules of each inbound and outbound transportation container are predetermined and known with certainty.

The objective of this research is to explore key design characteristics of a PI intermodal interface. The approach to be taken is to develop and use operational heuristics on different scenarios that embody some of the unique operating characteristics of the PI, notably completely decentralized and dynamic decision making regarding on the contents of outbound transportation containers. The contribution of this work is that it is the first investigation into this unique facility. It should be noted that the viewpoint of this work is to explore how unique operational features of the distributed PI intermodal interface
impact certain key parameters of the physical layout from a design perspective. This research is not an attempt at developing an optimal design or a methodology that will yield an operational design.

5.4.1 Notation

It is assumed that truck trailers arriving to the intermodal facility (i.e. inbound trailers) carrying unit loads that are destined for another location. This means that we are only considering unit loads and not any operation associated with modular containers (e.g., disaggregating and reaggregating, local delivery). The transportation inside the intermodal facility can be executed manually (e.g. by workers of using forklifts) or using an automated system (e.g. with a network of conveyor belts). It assumed that unit load are transported internally manually in material handling vehicles such as forklifts. Furthermore, the service mode of the intermodal facility is assumed to be exclusive where each dock door is dedicated to either an inbound or outbound containers. One side of the facility is assigned to inbound trucks and the other side to outbound trucks.

Let $I \geq 1$ be the number of inbound locations and $O \geq 1$ be the number of outbound locations. In this problem the planning horizon is limited to $T \geq 1$ time periods. If an inbound truck trailer from location $i; i = \{1, \ldots, I\}$ carries $n$ unit loads destined to location $j; j = \{1, \ldots, O\}$ in time period $t; t = \{1, \ldots, T\}$, we denote it by $n_{tij}$. Outbound capacity of all truck trailers is known at the beginning of the time period. Outbound capacity of a truck trailer to location $j$ in time period $t$ is denoted by $o^t_j$. Duration of a time period $t$ is $T_{\text{max}}$ minutes.

Unit loads in inbound truck trailers need to be moved to outbound truck trailers. If a unit load destined to location $j$ cannot be loaded into a truck trailer destined to location $j$, it can be stored temporarily at the intermodal facility. We assume that it costs $s$ to store a unit load for one time period. In addition, the intermodal facility has a maximum storage capacity of $S$ units at a given time.

Alternatively, if a unit load is destined to location $j$ and it cannot be loaded into
a truck trailer destined to location \( j \), it can be shipped to another destination \( k; k = \{1, \ldots, O\}, k \neq j \) where it will be routed to the final destination \( j \). Reshipping a unit load destined to \( j \) to an alternate destination \( k \) costs \( \$p_{jk} \) per unit load. It is assumed that, in addition to scheduled outbound shipments, additional shipments can be arranged to a specific location (e.g., an TL shipment can be arranged from the intermodal hub directly to any other intermodal hub) at a fixed cost of \( \$x \) per created shipment and a variable cost per unit load. For simplicity, per unit variable cost for additional shipment is assumed to be equivalent to the unit load storage cost \( (s) \). Thus, cost for shipping \( u \) unit loads in an additional shipment is \( x + u \cdot s \). It should be noted that additional shipment is the last resort option and it is analogous to storing the unshipped capacity elsewhere at an additional cost.

Intermodal facilities will undoubtedly have many different layouts to accommodate the types of operations that must be accommodated including varying amounts of temporary storage, space and material handling to disaggregate and re-aggregate transportation containers and unit loads. This will also extend to the number and arrangement of the dock doors based on volume and operations. In this research the simplest possible intermodal facility layout is assumed as illustrated in Figure 5.1. This resembles the common I-shaped cross-docking facilities which is ideal for cross-docking facilities with 150 doors or less [25]. It is also assumed that the intermodal facility has \( L(\geq I) \) inbound dock doors and \( M(\geq O) \) outbound dock doors. The distance between dock doors, \( d_{lm} \) is known. Material handling vehicles, such as fork lifts, move the unit loads in the intermodal facility. It is also assumed that there are \( v \) number of identical material handling vehicles. Each vehicle can carry a maximum of \( U(\geq 1) \) unit loads per trip and they can travel \( s_f \) feet per minute when they’re not loaded and \( s_b \) when they are loaded. Time to pick a unit load from an inbound truck trailer is \( t_p \) minutes and time to store a unit load in an outbound truck trailer is \( t_s \) minutes. When more than one unit load is stored or picked, it is assumed that it takes less time to handle them than handling them individually. Similar to the assumptions in Chapter 4, time for storing \( u \) unit loads is \( u \cdot t^c_s \) where \( c \) is a constant less
than 1 and time for picking \( u \) unit loads is \( u \cdot t_p^c \).

![Diagram](image)

Figure 5.1: A single-stage intermodal interface layout (adapted from Belle et al. [120]).

### 5.4.2 Assumptions

The following additional assumptions are made:

- All inbound truck trailers arrive at the beginning of each time period.
- All outbound truck trailers leave at the end of each time period.
- The details (e.g., destination and load or capacity) of inbound and outbound shipments in future time periods are not known. Details of shipments for the current time period is known at the beginning of the time period.
- Unit loads can be stacked on each other while being transported.
- All material handling vehicles work independently.
- When unit loads are temporarily stored, handling and traveling are instantaneous with no time for these activities included in the model.
- When an extra outbound shipment is created, it is done separately from normal operations and does not affect the time or vehicle utilization restrictions.
- Pre-emption is allowed. Unloading or loading of a truck trailer would be interrupted at the end of the time period.
5.5 Technical Approach

Since the focus of this study is to explore the relationships between performance measures and facility features, a construction heuristic was developed to solve the above problem. Routes between inbound dock door locations and outbound dock doors and storing decisions are constructed with the algorithm given below. The objective of the algorithm is to minimize the penalty for reshipping and storing by moving the outbound demand to outbound trailers within the time frame. The detailed pseudocode is given in Algorithm 9 in Appendix A.

**Step 1.** Assign an inbound dock door to each inbound truck randomly.

**Step 2.** If there is an outbound truck to a location, assign an outbound dock door.

**Step 3.** Transport direct shipments at inbound dock door \( l \). If no direct shipments can be assigned, mark as Unshipped. Repeat for all occupied inbound dock doors and go to Step 4. If Elapsed Time exceeds the time limit mark as Unshipped and go to Step 10.

**Step 4.** If there is Unshipped demand to destination \( j \) at inbound dock door \( l \), find alternate destination \( k \) with minimum reship cost.

**Step 5.** If reship cost to \( k \) is less than store cost, transport demand to outbound dock door where outbound truck to \( k \) is located if capacity to \( k \) allows. Otherwise mark as Unshipped. If Elapsed Time exceeds the time limit, go to Step 10.

**Step 6.** If store cost is less than reship cost, store the Unshipped demand destined to \( j \). If storage capacity is exceeded and go to Step 7.

**Step 7.** Transport Unshipped demand with alternate destination \( k \) to outbound dock door where outbound truck to \( k \) is located if capacity to \( k \) allows. If Elapsed Time exceeds the time limit, go to Step 10. If no alternate destinations can be found, go to Step 10.

**Step 8.** \( j \leftarrow j + 1 \) and go to Step 4. If all Unshipped demand is satisfied, go to Step 9.

**Step 9.** \( l \leftarrow l + 1 \) and go to Step 4. If all inbound dock doors are visited, go to Step 10.

**Step 10.** If there is Unshipped demand at dock door \( l \) to destination \( j \), create an additional shipment to location \( j \). Repeat for all inbound dock doors.

**Step 11.** Calculate Total Cost for iteration \( y \).

**Step 12.** If Total cost is less than \( C_{\text{min}} \) update BestRoute.
**Step 13.** Return to Step 1 until the maximum number of iterations is satisfied. At the end of iterations, select *BestRoute* as the solution for time period \( t \).

**Step 14.** Update the outbound demand list of \( t+1 \) with *Stored* demand.

**Step 15.** \( t \leftarrow t + 1 \) and return to Step 1. If time period \( t \) is last planning time horizon, stop.

There are three major phases in this algorithm: 1.) Assigning inbound and outbound dock doors, 2.) Determining direct outbound shipment assignments, 3.) Decide action for unshipped unit loads. First, inbound dock doors are assigned to inbound truck trailers randomly. If there is an outbound truck to location, \( j \) then the inbound dock door \( l \) with largest outbound demand to location \( j \) is identified. Then, an unused outbound dock door \( m \) is identified which is closest to the inbound dock door \( l \) (i.e. available \( m \) with minimum \( d_{lm} \)) and assign outbound trailer \( o^l_j \) to outbound dock door \( m \). If there is an outbound trailer with no direct shipments to that destination, a random unused outbound dock door is assigned.

Once inbound and outbound dock doors are assigned, the next step is to assign direct outbound shipments. When there is an outbound demand to destination \( j \) and there is an outbound trailer to destination \( j \) at the same time period, a direct outbound shipment can be assigned. An internal route is generated to transport the demand \( n^l_{ij} \) from inbound dock door to outbound dock door. If the available capacity of the outbound trailer \( o^l_j \) is less than the demand, partial outbound demand is transported while remaining portion is considered as *Unshipped* \( ij \). If there is no direct outbound trailers, that demand is also considered as *Unshipped* \( ij \) demand. Repeat this procedure for each trailer at inbound dock doors. In case the total time for direct shipment assignment exceeds the time limit of the period \( T_{max} \), remaining unassigned outbound demand is also considered as *Unshipped* \( ij \).

After direct outbound shipments are assigned, the remaining unit loads can be stored or reshipped to alternate destinations. If there are unshipped unit loads after the storage operation, an alternate shipping location, \( k \) for each *Unshipped* \( ij \) is found. Alternate shipping location \( k \) should have an outbound shipment at the current time...
period and there should be excess capacity available in the trailer. Also, the alternate destination cost \( p_{jk} \) for each \( k \) should be the minimum among the available outbound destination. Then, shipping cost for each alternate shipment is calculated. If the alternate shipping cost is less than storage cost for a given \( Unshipped_{ij} \), then the internal routes for alternate destinations are generated similar to direct shipment assigning. In the event where no alternate shipping is possible and storage capacity is fully utilized, or if routing time exceeds the time limit of the period, additional shipments are arranged at a cost of \( $x \) per created shipment per location.

If there is more than one material handling vehicle, it is assumed that the vehicles perform independently. For example, when there are two trips and one vehicle, the vehicle has to finish the first trip before beginning the second trip. If there are two vehicles, they can work simultaneously and finish both trips in half the time of the single vehicle case. Therefore, the time limit of a period depends on the available number of vehicles.

These three steps are repeated for a pre-determined number of iterations (\( y_{max} \)) with a different set of random inbound dock doors in each iteration. At the end of the iterations, the set of inbound and outbound dock doors, storage quantity or alternate shipping destinations, the quantity for each unshipped outbound demand, and any additional shipments that gives lowest total cost for time period \( t \) is recorded. The stored unit loads at time period \( t \) are considered as outbound demand for time period \( t + 1 \) in addition to the outbound demands of next time period (\( n_{ij}^{t+1} \)). Unused storage in time period \( t \) is considered as the available storage quantity in time period \( t + 1 \).

This is repeated for each time period \( t \) until the end of planning time horizon \( T \). At the end of the time horizon, the set of best solutions for each time period \( t \) is considered as the solution for the problem and statistical data such as total cost, maximum stored units, and total reshipped units are recorded.

The dock door assignment problem has been studied extensively in literature; however, in this research the focus is on utilizing the characteristics of the PI into the operations of an intermodal interface. Therefore, dock door assignment is not considered
as a separate problem and in this research the dock door assignment uses a common sense approach where inbound dock doors are assigned randomly and outbound dock doors are assigned based on the availability of outbound trailers and distance between dock doors.

Similarly, the pick & delivery vehicle routing problem can be considered a separate optimization problem to improve routing by methods such as combining routes, etc.. Here, we approach this with a simple single-command delivery approach is used to transport outbound demand to outbound truck trailers. The vehicle transports unit loads from an inbound dock door \( l \) to outbound dock door \( m \) until all the assigned demand is delivered. The vehicle would not visit other dock doors in the same trip. Once all outbound demand in an inbound dock door is delivered, the operation is repeated starting with the adjacent occupied inbound dock door.
5.6 Scenarios and Results

Several problem instances were solved to using the above algorithm validate the model and study the intermodal facility performance with differing values of key parameters. The performance parameters to be investigated are the maximum number of units stored in a given period, the total number of unit loads reshipped, total number of trips, total cost, and the total operation time.

The maximum number of unit loads stored in a given period, which can be calculated if there is no upper bound in storage limit ($S$) is important to understand the storage requirements. The ability to ship unit loads to alternative destinations is a key parameter in the PI and number of unit loads reshipped is a measure of using that option under different scenarios. The number of unit loads shipped to alternative destinations over the planning time horizon is used in this research as a performance parameter. The total cost for a given problem is the storage cost and reshipping cost aggregated over the planning horizon. For example, if $a$ number of unit loads are stored for one time period, $b$ unit loads are reshipped to alternate destination $k$ instead of original destination $j$, and $c$ unit loads are shipped to location $l$ as an extra outbound shipment, total cost for the time period is $a \cdot s + b \cdot p_{jk} + (x + c \cdot s)$. The total operation time is the total of elapsed time while transporting unit loads from inbound dock doors to the outbound dock doors; this includes travel time and handle time. Total operation time is an indication of the utilization of unit load handling vehicles. The total number of trips is another indication of the utilization of resources. It is the total number of tours taken by each vehicle to deliver outbound demand for each time period aggregated over the time horizon.

Once again, it should be noted that the scale of these examples are likely not the size of the real-world intermodal interfaces; however, the point of this work is to explore fundamental relationships and these problem sizes allow this type of analysis to be performed.
5.6.1 Unlimited outbound capacity

This first scenario explores how material handling capacity influences intermodal facility performance. This scenario assumes that outbound truck trailer capacity exceeds the outbound demand so there are enough resources to move the unit loads in outbound truck trailers within the given time frame. On the other hand, material handling capacity, represented here by the number of vehicles available to move the freight within the intermodal facility, is limited and represent the parameter being investigated.

In this scenario, the number of unit load handling vehicles is varied and the impact on the performance measures is monitored. Note that with unlimited outbound capacity, these performance measures are strictly dependent on material handling capacity because neither storage nor reshipping is required due lack of outbound capacity as will be the case in later scenarios. Here, we assume that there are 20 inbound dock doors \((L)\) and 20 outbound dock doors \((M)\) in the intermodal facility. There are 20 inbound locations \((I)\) and 20 outbound destination locations \((O)\). Trucks arrive to the 20 inbound locations in each time period carrying a random number of unit loads that can vary between 0 and 30 in multiples of 5. In other words, a uniform distribution between 0 and 6 is multiplied by 5 to represent the capacity of each inbound container. It is assumed that the unit loads in each inbound container is destined to only two outbound locations. For example, if the inbound capacity from inbound location \(i\) is \(z\) unit loads, a random amount \(z_1 \leq z\) would have final destination \(j_1\) and the remaining amount, \(z - z_q\), would have another final destination, \(j_2\). The final destinations \((j_1, j_2)\) for each inbound truck in each period are assigned randomly.

To represent unrestricted outbound capacity, it is assumed that each outbound truck has a capacity of 100 unit loads, which is greater than any inbound truck capacity. Maximum storage capacity \((S)\) is set to 10,000 units which leaves the option to store the unshipped unit loads if material handling vehicles do not have enough resources to move the outbound demand within the intermodal facility. Storage cost per unit load per period \((s)\) is assumed to be $2. The cost to reship one unit load to another location
\( p_{jk} \) is assumed to be distributed uniformly between $1 and $5 for each location pair and it does not change throughout the problem. This makes the average reshipping cost per unit load more costly than the storing cost per unit load. Although it is cheaper to store the unshipped unit loads, it might not be able to store all the unit loads if the storage capacity of the intermodal interface is limited. Also, unit loads might need to be stored for more than one time period which would cost more in the end compared to reshipping them. On the other hand, reshipping the unshipped unit loads as the first choice would lead to delays and complex routes; thus, reshipping costs more than storing in general. This presents an interesting operational problem for handling the unshipped unit loads.

The maximum time (\( T_{\text{max}} \)) to handle moving unit loads is assumed to be \( 60 \) minutes and the total planning horizon (\( T \)) is assumed to be \( 20 \) time periods. To reflect current handling operations, especially with unit loads in the PI currently envisioned to be structurally rigid cubes, it is assumed that unit loads are handled by vehicles which can carry up to 2 in a trip \( U \). To investigate the sensitivity to material handling capacity, the number of vehicles is varied from 4 to 16. The unloaded speed (\( s_f \)) and the loaded speed (\( s_b \)) of vehicles are set to \( 300 \) ft/min. Time to pick a unit load from an inbound truck trailer (\( t_p \)) and the time to store a unit load in an outbound truck trailer (\( t_s \)) is assumed to be \( 0.3 \) minutes. The multiple unit load handling constant \( c \) is set to \( 0.9 \) and the number of iterations (\( y_{\text{max}} \)) is limited to \( 50 \).

Figure 5.2 shows the performance measures: Total time, Total number of trips, Maximum number of stored unit loads, Total number of reshipped unit loads, and Total cost for number of vehicle variation with unlimited outbound capacity and unlimited storage. Total time is in minutes; Total cost is in dollars. The performance measures are averaged for 30 problem instances.

It is observed from Figure 5.2 that the total number of reshipped unit loads is zero irrespective of the number of vehicles used. This observation is expected due to availability of a large storage capacity and the outbound capacity to each destination exceeding the outbound demand in each time period. When there are enough material
handle the material handling vehicles, all outbound demand can be moved to outbound trailers within the time frame and no reshipping would be necessary. On the other hand, when there is not enough material handling vehicles to move the freight within the intermodal facility, the unit loads can be stored in the facility, which costs less than reshipping. As a result, no unit loads require reshipping.

Cumulative number of trips and time of trips by all vehicles increase linearly to the number of vehicles until they reach a plateau. In this scenario, when number of vehicles is 12 or more, total time and total trips reach the plateau. When the number of vehicles is low, all the outbound demand cannot be delivered to the outbound containers and the unshipped unit loads are stored, which does not require the material handling vehicle resources. As a result, number of trips is limited by the available number of vehicles. Total route time is proportional to the total number of trips. However, the ratio between total time and total number of trips gradually increases with the number of vehicles. In other words, time for each trip increases with the number of vehicles. For example, the ratio between total time and total trips is 1.97 for 4 vehicles case and it becomes 2.15 for 16 vehicle case. When there is limited number of vehicles, some portion of the unit loads
are stored which does not include traveling. On the other hand, when there is plenty of material handling resources, all unit loads are moved to the outbound dock doors and therefore more distance is traveled. As a result, travel distance per trip increases when there are more material handling vehicles.

It is evident from the results that the maximum number of unit loads stored at the end of time horizon is inversely proportional to the number of vehicles. When the cumulative time to move the unit loads to outbound truck trailers exceeds the time limit, unit loads that are not loaded are stored until the next period. When the demand for freight handling is more than the material handling capacity of vehicles within the time frame, storage accumulates and maximum number of unit loads stored increases in each time period. An increased number of vehicles means more unit loads can be moved to outbound trailers within the time limit. If all inbound unit loads can be moved to outbound trailers, the number stored is zero. Total cost is also inversely proportional to the number of vehicles, as storage cost is the only cost that occurs in this problem. However, it should be noted that storage cost is calculated for cumulative number of unit loads stored per period over time horizon while maximum number of stored unit loads is an upper bound of storage requirement.

In addition to the average parameters, minimum and maximum values of total time, total number of trips, and total cost for number of vehicles variation is shown in Figure 5.3. It is evident that total time, which is an indicator of utilization of material handling vehicles, show a small variation when number of vehicles is low. This suggests that vehicles are utilized at their full capacity for each test problem when number of vehicles is low. Vehicles are fully utilized which is as a result of lack of resources, and thus natural variation of the characteristics of the problem are suppressed. On the other hand, when number of vehicles are increased, the variation of total time increases. With more material handling resources available, differences in characteristics of each problem instance becomes apparent in the results. Total number of trips show similar variation; when number of vehicles is low, the variation of total number of trips is narrow while it
<table>
<thead>
<tr>
<th>Number of vehicles</th>
<th>Total Time (min)</th>
<th>Total Trips</th>
<th>Total Cost ($)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>2,000,000</td>
<td>40,000</td>
<td>10,000</td>
</tr>
<tr>
<td>6</td>
<td>4,000,000</td>
<td>60,000</td>
<td>20,000</td>
</tr>
<tr>
<td>8</td>
<td>6,000,000</td>
<td>80,000</td>
<td>30,000</td>
</tr>
<tr>
<td>10</td>
<td>8,000,000</td>
<td>100,000</td>
<td>40,000</td>
</tr>
<tr>
<td>12</td>
<td>10,000,000</td>
<td>120,000</td>
<td>50,000</td>
</tr>
</tbody>
</table>

Figure 5.3: Minimum and maximum time, number of trips, and cost vs. number of vehicles

increases when number of vehicles is increased. The number of trips is also an indicator of utilization of vehicles. This shows that number of vehicles is a bottleneck for the problem.

Total cost variation is higher when number of vehicles is low. Depending on the test problem’s characteristics, usage of storage space varies for when number of vehicles is now. When there is sufficient material handling capacity, need of storage becomes zero for all test problems since the outbound capacity exceeds the outbound demand. As a result, the variation of total cost decreases when number of vehicles is increased.

These results are consistent with intuition so it confirms the heuristic is performing as expected. Furthermore, this type of chart (Figure 5.2) can be used in facility operations to estimate number of vehicles required during a period if the storage level is given or it could be used in design to estimate the storage required if for a given number of vehicles. Of course, the strong assumption here is that outbound capacity exceeds outbound demand.
5.6.2 Limited outbound capacity

Clearly, there will be time periods when outbound capacity does not exceed outbound demand in an inter modal facility and that is the situation to be explored here. It is important, however, to note that the average outbound capacity over time must be greater than or equal to the average inbound demand or inventory levels in the facility would grow without bound as in a simple queuing model with intensity greater than one. We model this using the ratio of outbound capacity to inbound load (referred henceforth as ”outbound/inbound capacity ratio”). In this scenario, the outbound/inbound capacity ratio is varied from 1.25 to 2.05 in increments of 0.2 and the number of unit load handling vehicles is varied for each. Inbound capacities were not changed from Section 5.6.1 which is a uniform distribution between 0 and 6 is multiplied by 5. Outbound capacities to each destination in each period were changed such that the outbound/inbound capacity ratio is the desired ratio. For example, uniform distribution between 0 and 7 multiplied by 5 gives an outbound/ inbound capacity ratio of 1.25 and a combination of a uniform distribution between 0 and 8 and a uniform distribution between gives an outbound/ inbound ratio of 1.45. It should be noted that the outbound/inbound capacity ratio is an approximate value. 10 test problem instances for each outbound/inbound capacity ratio are used in this scenario.

If the resources allow and if reshipping is cheaper than storage unit loads can be reshipped to final destination via a different location. It was assumed that the intermodal facility has enough storage to accommodate the unit loads not (directly or indirectly) shipped. The rest of the parameters were not changed from the scenario in Section 5.6.1. Figure 5.4 shows the maximum number of unit loads that were stored for the different outbound/ inbound capacity ratios when number of vehicles are varied. The maximum number of units stored over time for a test problem with outbound/inbound ratio of 1.25 for a single iteration is shown in Figure 5.5.

Figure 5.4 shows that, similar to the case with unlimited outbound capacity, maximum storage capacity is inversely proportional to the number of vehicles. When number
of vehicles is low, unshipped unit loads are stored in the intermodal interface because reshipping to alternate locations also cannot be handled due to inadequate material handling capacity. Creating extra shipments costs more than storing the unit loads when decision is made considering only one time period. However, lack of material handling vehicles lead the storage to accumulate and number of unit loads stored increases in each time period. This is evident by the increase of maximum number of stored units over time in Figure 5.5 where maximum number of stored units increases in each time period when number of vehicles is 6 or less. When number of vehicles is 8 or above, the maximum number of stored units plateaus after several time periods. That means, when number of vehicles is 8 or more material handling demand can be satisfied without storage accumulation. Since the maximum number of unit loads stay unchanged when there are 8 or more vehicles, it can be deducted that at least 8 vehicles are required to handle internal unit load shipping. This is an example of using the model for planning purposes.

In addition, maximum required storage does not reach zero even with use of large number of vehicles. When outbound/inbound capacity ratio is higher, more units can
be directly shipped instead of storing or reshipping. Therefore, for a given number of vehicles, less storage is required. Because outbound capacity might not always exceed outbound demand and on reshipping is more costly than storing, some unit loads might be stored at the intermodal facility. As a result, even when resources to move unit loads is not a limitation, some storage is required because of the strategy this heuristic schedules shipments.

Also, required maximum storage capacity for a given number of vehicles is slightly smaller when the outbound/inbound capacity ratio is higher. This is to be expected since with more outbound capacity, more direct shipments is possible and need of storage is less. The maximum storage capacity reduction when outbound/inbound capacity ratio is increased when number of vehicles is 4 is 1.34%. However, when number of vehicles is increased, i.e. when there is enough material handling resources, maximum storage capacity reduction is 57.2% when outbound/inbound capacity ratio changes from 1.25 to 2.05 for 10 vehicles. This suggests that outbound/inbound capacity ratio is significant only if there is enough material handling capacity.

Total cost for different outbound/inbound capacity ratios when number of vehicles
Figure 5.6: Total cost vs. number of vehicles for different outbound/inbound capacity ratios.

is varied is shown in Figure 5.6. Table 5.1 shows the total number of reshipped unit loads for limited outbound capacity scenario. Total cost variation is similar to the maximum number of stored unit loads variation: both are inversely proportional to the number of vehicles and for a given number of vehicles increase in outbound/inbound capacity ratio reduces the value. When number of vehicles is low, priority is given to direct outbound requests and the number of reshipped units remains zero until all direct outbound requests are fulfilled, as seen from the data in Table 5.1. As a result the storage cost is the only contribution to the total cost when number of vehicles is low.

When more vehicles are available, unit loads that cannot be directly shipped can be reshipped if it costs less than storing at the intermodal facility. Therefore, the number of reshipped units for a given outbound/inbound capacity ratio increases with number of vehicles as the data suggests. When there is adequate material handling capacity, storage does not accumulate over time and as a result total cost does remains low. Storage utilization is the major contributor to the total cost in this case too, because reshipping is done if it is cheaper than storing when there is no limit on storage capacity.

Another observation is that the number of reshipped units for a given outbound-
/inbound capacity ratio increases with number of vehicles and then it reaches a plateau. The initial increase of reshipped units is due to the availability of more material handling resources. Even with enough material handling capacity, if outbound capacity is limited reshipping is not possible and thus reshipped units reach a plateau. For a given number of vehicles, increase of outbound/inbound capacity ratio results in lower reshipping because more direct shipping is possible.

<table>
<thead>
<tr>
<th>Number of vehicles</th>
<th>Outbound/Inbound capacity ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.25</td>
</tr>
<tr>
<td>4</td>
<td>0.0</td>
</tr>
<tr>
<td>6</td>
<td>13.0</td>
</tr>
<tr>
<td>8</td>
<td>936.6</td>
</tr>
<tr>
<td>10</td>
<td>1,320.5</td>
</tr>
<tr>
<td>12</td>
<td>1,349.0</td>
</tr>
<tr>
<td>14</td>
<td>1,338.5</td>
</tr>
<tr>
<td>16</td>
<td>1,336.5</td>
</tr>
</tbody>
</table>

Table 5.1: Total number of reshipped unit loads for different outbound/inbound capacity ratios
5.6.3 Limited outbound capacity and limited storage

In this scenario, the more realistic constraint is added that storage capacity is a limited resource but all inbound freight must be handled within the restricted time limit. When the storage limit is exceeded and no direct or indirect outbound shipping is available, unit loads must be sent to destinations other than the correct one for that unit load at an additional cost. Exploring the performance under these restrictions is clearly important since it represents a much more realistic scenario with respect to design where the owner much balance capital expenditures for the initial facility against ongoing operations costs and the penalty costs associated with reshipping.

In this scenario, the storage limit is varied from 100 to 500 unit loads in increments of 100 for different outbound/inbound capacity ratios, 1.25 to 2.05 which were used in Section 5.6.2. Also, the number of vehicles used was varied from 4 to 10 in increments of 2. Maximum number of vehicles was limited to 10 because no significant differences between parameters were observed when number of vehicles is above 10. The set of 10 test problems used in Section 5.6.2 was used in this scenario as well.

Figure 5.7 shows the number of reshipped units for a given outbound/inbound capacity ratio for different number of vehicles when storage capacity is varied under limited outbound capacity and limited storage scenario. The number of reshipped units for a given number of vehicles for different outbound/inbound capacity ratio when storage capacity is varied is presented in Figure 5.8.
Figure 5.7: Number of reshipped units vs. storage limit for different number of vehicles for a given outbound/inbound ratio
For a given outbound/inbound capacity ratio and a storage capacity, increasing number of material handling vehicles from 4 to 6 results in reduction of reshipped units for all storage capacity limits. When number of vehicles is low, if direct outbound demand cannot be fulfilled within the time frame unshipped demand needs to either be stored or sent to the destinations in special shipments because reshipping to alternate destination needs material handling resources too. As a result, most of the unshipped unit loads would be shipped as special shipments because storage capacity is limited. Increase of material handling vehicle quantity from 4 to 6 enables to fulfill direct shipment demand.
instead of reshipping as special shipments and thus, a drop of number of reshipped units can be observed. However, small storage limit results in more reshipments than a larger storage limit. Increase of number of vehicles affect less towards reshipping for the higher storage capacity case compared to lower storage capacity case. Therefore, the drop of number of reshipped units gradually decreases. For example, the reduction of number reshipped units is 37.7% for storage limit of 100, 33.4% for storage limit of 200 and 29.7% for storage limit of 300.

For a given outbound/inbound capacity ratio, the number of reshipped units increases when number of vehicles is in the higher end, e.g. when number of vehicles is 10, compared to 8 vehicles case. If there is adequate number of material handling vehicles to handle both direct outbound shipments and reshipments to alternate destinations, reshipping option is taken if it costs less than storing the unshipped units. As a result, more reshipped units can be observed when number of vehicles is higher.

It is evident from the Figure 5.8 the for a given number of vehicles and a given storage limit, number of reshipped units decreases when outbound/inbound capacity ratio is increased. This is as a result of more direct shipments when outbound capacity is increased. It should be noted that when number of vehicles is 4, the number of reshipped units vary less than 3% of the minimum time when outbound/inbound capacity ratio is varied for a given storage capacity. However, when number of vehicles is 10, the number of reshipped units vary by about 45% between maximum and minimum values. This implies that although more outbound capacity enables more direct shipping, there should be adequate material handling resource to take advantage of outbound capacity.

When the number of vehicles is 4, the number of reshipped units decreases when storage limit is increased for a given outbound/inbound capacity ratio; The difference of number of reshipped units between 100 unit storage capacity and 500 unit storage capacity is 29.4% on average. However, when the number of vehicles is 10, the number of reshipped units for a given outbound/inbound capacity ratio varies less than 3% between maximum and minimum values. When there is limited material handling resources, un-
shipped unit loads can be either stored in the intermodal facility or reshipped and the less costly option is chosen. Since reshipping a unit load costs more than storing considering one time period, storage option is taken if storage capacity permits. As a result, lesser number of units are reshipped when storage limit is high. However, when there is adequate material handling resources reshipping is affected by the outbound capacity, not the storage capacity. Therefore, number of reshipped units does not vary less when storage capacity is varied.

Figure 5.9 shows the maximum number of stored units for each outbound/inbound ratio under limited outbound capacity and limited storage capacity scenario. Storage capacity is fully utilized when number of vehicles is low irrespective of the outbound/inbound capacity ratio. When number of vehicles is increased, utilization of storage capacity reduces. This is another example of the possibility of using the model as a planning tool to determine required material handling capacity and/or storage capacity.

Also the Figure 5.9 indicates that when number of vehicles is high, storage utilization decreases when outbound/inbound capacity ratio is increased. More direct shipping and reshipping to alternate destinations is enabled by higher outbound capacities and if material handling vehicles can handle the internal routing demand, less storage capacity is required for higher outbound/inbound ratios.

The total cost for a given outbound/inbound capacity ratio for different number of vehicles when storage limit is varied under limited outbound capacity and limited storage scenario is presented in Figure 5.10. The total cost for a given number of vehicles for different outbound/inbound capacity ratios when storage limit is varied under limited outbound ratio and limited storage capacity scenario is shown in Figure 5.11.
Figure 5.9: Maximum number of stored units vs. storage capacity limit for different number of vehicles for a given outbound/inbound capacity ratio.
Figure 5.10: Total cost vs. storage capacity limit for different number of vehicles for a given outbound/inbound capacity ratio.
Figure 5.11: Total cost vs. storage capacity limit for different outbound/inbound capacity ratios for a given number of vehicles

From the Figure 5.10, it is clear that for a given outbound/inbound capacity ratio and a given storage capacity, total cost decreases when number of vehicles is increased. When number of vehicles is increased from 4 to 6, number of reshipped units decreases as evident in the Figure 5.7. This directly affects the total cost and it, too, decreases. However, despite number of reshipped units increases when number of vehicles is increased, total cost decreases. The increase of reshipment when there is more vehicles results in less utilization of storage, as seen in the Figure 5.9. Reshipping to alternate locations is chosen only if it costs less than storing when there is unused storage capacity and therefore total
cost decreases when number of vehicles is high.

An interesting observation is that for a given outbound/inbound ratio, total cost increases with storage capacity limit when number of vehicles is low, as can be seen in Figure 5.11. For example, total cost increases by 27.7% when storage capacity is increased from 100 to 500 unit loads when outbound/inbound ratio is 1.25 and number of vehicles is 4 while total cost increase for 10 vehicle case is 0.7%. This is as a result of making storage vs. reship decisions with demand and capacity information for current time period only. When there is unused storage capacity available, the algorithm analyzes the immediate storage cost vs. reship cost to make a decision to select storing or reshipping. However, this results in handling more unit loads in the next time window. This actually leads to a vicious circle of storage continuing to accumulate and thus, total the cost goes up when there is more storage available. However, with more material handling vehicles, storage accumulation does not occur and therefore total cost variation is less than 2% for each outbound/inbound capacity ratio.

The total cost decreases approximately by 1% when outbound/inbound ratio is increased from 1.25 to 2.05 for each storage when number of vehicles is 4. On the other hand, when number of vehicles is 10, total cost reduces by at least 56% when outbound/inbound ratio is increased from 1.25 to 2.05. This also suggests that increased outbound capacity is effective only if there is enough material handling capacity available.

5.6.4 Sensitivity to storage cost

In above scenarios, storage cost was considered a constant of $2. To analyze the sensitivity of storage cost, storage cost was varied from $0.5 to $3 in intervals of $0.5 for different outbound/inbound capacity ratios and number of vehicles with limited outbound capacity and limited storage capacity assumptions. Outbound/inbound capacity ratios were selected to be 1.25 to 2.05 in increments of 0.2, which were used in Section 5.6.2. Storage capacity limit was set at 500 units to enable storing if required. Same test of 10 test problems used in Section 5.6.2 was used in this scenario as well. Figure 5.12 shows
maximum number of stored units vs. storage cost for a given number of vehicles under limited outbound/inbound capacity and limited storage capacity scenario. Total number of reshipped units vs. storage cost for each number of vehicles is shown in Figure 5.13. Figure 5.14 shows the total cost vs. storage cost for a given number of vehicles for the same scenario.

From Figure 5.12, it is clear that when number of vehicles is low, i.e. when material handling resources is inadequate, 100% of the storage capacity is utilized irrespective of the storage cost. Furthermore, variation of number of reshipped units when number of vehicles are low is small which can be observed from Figure 5.13. For example number of reshipped units show less than 4% variation between maximum and minimum values for a given outbound/inbound capacity ratio for 4 vehicle and 6 vehicle cases. Therefore, it is clear that unit load storage cost per period has no effect on store and reship decisions when material handling resources are limited.

When number of vehicles is sufficient to fulfill the unit load handling demand, maximum storage capacity is not utilized. In that case, the storage capacity utilization decreases when the outbound/inbound capacity ratio is increased, irrespective of the storage cost. Similarly, number of reshipped units also decreases when outbound/inbound capacity ratio is increased, irrespective of the storage cost. However, when number of vehicles is high, the storage capacity utilization for a given outbound/inbound capacity ratio decreases when storage cost is increased throughout the storage cost range. On the other hand, the number of reshipped units for a given outbound/inbound capacity ratio increases when reship cost is increased. This suggests that when storage cost is increased, the algorithm chooses to reship unshipped unit loads instead of store them for a given outbound/inbound capacity ratio. This implies that unit load storage cost when material handling resources are sufficient, store and reship decisions are affected by the storage cost.

Although the storage capacity utilization decreases and reshipping increases for a given outbound/inbound ratio when storage cost is increased, some exceptions can be
Figure 5.12: Maximum stored units vs. storage cost for different outbound/inbound capacity ratios for a given number of vehicles
Figure 5.13: Number of reshipped units vs. storage cost for different outbound/inbound capacity ratios for a given number of vehicles
Figure 5.14: Total cost vs. storage cost for different outbound/inbound capacity ratios for a given number of vehicles.
observed. For example, when number of vehicles is 8 and outbound/inbound capacity ratio is 1.45, minimum storage utilization is recorded when unit load storage cost is $2 despite minimum storage utilization is recorded for $3 storage cost, in general. This suggests that storage cost can affect the reship and storage decisions based on the characteristics of the problem. However, these exceptions alone do not change the final objective, which is the total cost.

As evident from Figure 5.14, total cost increases for a given outbound/inbound capacity ratio when unit load storage cost is increased. This is as a result of the increased storage cost component of the total cost. Nevertheless, it’s apparent that reshipping cost, which is the other component of the total cost is unaffected by the variation of storage cost.

When number of vehicles is low, the variation of total cost between different outbound/inbound capacity ratios is small. The variation of total cost for a given storage cost is less than 4% when number of vehicles is 4 or 6. Increased outbound capacity cannot be utilized when material handling resources are inadequate. When number of vehicles is increased, increased outbound capacity results in reduction of total cost throughout the unit load store cost range. In conclusion, storage cost affects the reship and store decisions only when there is adequate material handling resources.

### 5.6.5 Sensitivity to reship cost

Reship cost for a destination pair ($p_{jk}$) was assumed to be a uniformly distributed value between $1 and $5 per unit load and reship costs were not changed in above scenarios. Reship cost ranges were varied in this scenario to analyze the sensitivity to reship cost. Uniform distribution between $1 and $2, $1 and $3, $1 and $4, and $1 and $5 were used as reship costs. Average reship cost for each range is $1.45, $1.95, $2.45, and $2.95 respectively. Storage capacity of the intermodal facility was set to 500 unit loads because it enables a choice between storing and reshipping if material handling capacity allows. An outbound/inbound capacity ratio of 1.65 was used for the ten test problems in this
scenario. Number of vehicles was varied from 4 to 10 in increments of 2. Figure 5.15 shows the total number of reshipped units for each reship cost range. The total cost for each reship cost range for each number of vehicles is given in Figure 5.16.

The date in the Figure 5.15 shows that total number of reshipped units decreases when number of vehicles is increased from 4 to 6, and 6 to 8, and it increases when number
of vehicles is increased from 8 to 10 for all reship cost ranges. The decrease of number of reshipped units is due to increase in material handling capacity and thus possibility of handling more direct shipments instead of creating special shipments when storage capacity is utilized. The increase of reshipment when number of vehicles is 10 is due to reshipping to alternate destinations with added material handling capacity. The decrease and increase of number of reshipped units does not change depending on the reship cost range.

Furthermore it can be observed that when number of vehicles is low, i.e. 4 and 6 vehicles case, total number of reshipped units show less than 2.5% variation between maximum and minimum values. When number of vehicles is 8 and 10, in other words when material handling capacity is high, the total number of reshipped units vary between by 9.3% and 5.4$ respectively. This implies that reship decisions are affected when number of vehicles are high.

Total cost for decreases when number of vehicles is increased for all reship cost ranges. However, the total cost for a given number of vehicles show little variation throughout the reship cost range. For example, difference in maximum and minimum total cost for reship cost range when number of vehicles is 4 and 6 are 0.5% and 0.7% respectively. The same when number of vehicles is 8 and 10 are 10.5% and 5% respectively. Similar to the number of reshipped units variation, reship cost affects total cost when the number of vehicles is high. In general, reship cost has more effect on the reship and store decisions when material handling capacity is high compared to material handling resources constricted scenario.

It should be noted that when there is inadequate reshipping options and no available storage capacity, unshipped unit loads in each time period are sent to destinations as special shipments. Since that is the last resort option, variation in extra reship cost does not affect the decisions. Therefore sensitivity of special shipment cost is not studied.
5.7 Conclusions and Future Work

In this research an operational algorithm was developed to determine internal unit load handling operations of an intermodal interface. A construction based heuristic was used to design routes and determine dock door locations for inbound and outbound transportation containers. Optimizing dock door assignment or optimizing internal unit load handling routes was not considered at this stage of the research. Furthermore, the operational algorithm was used as a tool to study how different parameters of intermodal interfaces affect internal operations. It was evident that number of material handling vehicles affect routing and storage decisions. When outbound capacity is limited and storage capacity is limited, it is important to have adequate material handling capacity. More outbound capacity is not usable if material handling capacity is insufficient. When there is adequate material handling capacity, storage limit or outbound capacity has less impact on the routing decisions.

While this research addresses some primary operational questions, further research is suggested to improve the internal operations of intermodal interfaces. The solutions from a construction heuristic can be far from optimum solutions. Metaheuristic methods may be used to improve the solutions while using the solutions from construction heuristic developed in this research as a starting point.

Outbound packages usually have a delivery deadline. As a result, some packages might need to be prioritized when deciding outbound loading assignments. Furthermore, having a delivery deadline prevents long-term storing and re-routing options. Therefore, including delivery deadlines for packages makes internal operations of intermodal interfaces complex while making it closer to reality.

 Interruption of operations is unavoidable. How disruptions of inbound transportation or outbound transportation such as delays and internal operation disruptions such as vehicle unavailability and storage limitation affect the decisions of intermodal interfaces is an interesting research question that needed to be addressed.
The operational algorithm developed in this research assumes that future inbound and outbound capacities are not known while making decisions for a given time period. Having information on inbound and outbound capacities of future time periods can affect decisions made at intermodal interface. Further research is required to incorporate that information into decision making at intermodal interfaces.
Chapter 6

Conclusions

6.1 Time-Based Multi-Command Operations in Unit Load Warehouses

Order picking and storing in storage facilities are often modeled using travel distance based network optimization methods. Such models often ignore a crucial element of order picking and storing operations: package handling time. Furthermore, ability to carry more than one unit load in a vehicle opens up new possibilities in routing material handling vehicles.

In the first part of this research, the effects of carrying two unit loads in a trip is studied. Routing in a unit load handling storage facility where one unit load can be stored at a location, i.e. with one-to-one mapping between locations and unit loads, multi-command operations where material handling vehicles store and/ or pick multiple unit loads was explored. A mixed-integer linear programming model that minimizes total operation time (travel time and unit load handling time) was formulated and solved for smaller problems. Sensitivity to different parameters of the problem was studied. Three route construction heuristics were developed to solve the problem for larger problem instances. Route construction heuristics provided solutions within 5% of the optimum solutions in a short amount of computing time. Future work for this chapter includes
considering random storage and dynamic store and pick demand for routing decisions, prioritize storage and picking for time windows, and improving the solutions methods using meta-heuristic methods.

Contributions of Chapter 3 are: formulating a mixed-integer linear programming model that minimizes total operating time to find optimal pick or store routes in unit load storage facilities with one-to-one mapping between locations where vehicles can carry more than one unit load in a trip, including multiple unit carrying ability to pick and store routing decisions, and developing route construction heuristics for the problem to solve larger problem instances.

6.2 Multi-Command Routing Operations in Unit Load Storage Facilities

In Chapter 4 routing in unit load storage facilities with no one-to-one mapping between locations and unit loads is addressed. In addition, possibility to carry more than two unit loads in a vehicle is considered. To solve the problem, a mixed-integer linear programming model which minimizes total travel and unit load handling time is presented. Four heuristics are introduced to construct routes in store and pick unit loads. First heuristic is based on visiting locations that uses the vehicle capacity as much as possible; Second heuristic visits the locations in the order of distance to the depot; Third heuristic visits the locations in the ascending order of operation time of each location; Fourth heuristic constructs routes by visiting the locations in descending order of the operation time of each location. The fourth heuristic provides overall better routes for the test problems, although distance based heuristic, also, provides similar results in most of the scenarios. Applying evolutionary algorithms to find better solutions, collaborating multiple vehicles for routing, routing with dynamic store & pick demand lists, and adding grouping restrictions to unit loads are among the possible future work for this chapter.

Contributions of Chapter 4 are: introducing routing with multiple unit load carry-
ing vehicles in order picking and storage warehouses, formulating a mixed-integer linear
programming model to determine store and pick routes in a storage facility with multiple
unit loads stored at a location that might need multiple visits to fulfill the demand, and
introducing heuristic methods to construct routes for the above problem.

6.3 Intermodal Interface Operations with Time Win-
dows and Reshipping

Chapter 5 studies internal operations in intermodal interfaces where trucks bring
unit loads destined to various locations in each time period. The unit loads need to
be transported to outbound trailers destined to final destination or an alternate desti-
nation within the time period or stored temporarily, usually until an outbound trailer
with enough capacity is available. An operational algorithm to determine internal unit
load handling operations was developed. The operational algorithm was used to study
how the intermodal interfaces are affected by parameters of the facility. The observations
suggested that material handling capacity, i.e. number of vehicles is important factor
for internal operation decisions. Modeling the intermodal interfaces which have multiple
goals is a required next step. The operational decisions can be improved with the use of
metaheuristic methods. Also including priorities for outbound shipments, using informa-
tion on inbound and outbound truck capacities in future time periods for decision making,
and considering service interruptions into operation decision are some topics for future
research.

Main contributions of Chapter 5 are: developing an algorithm to make internal
operation decisions in an intermodal interface where reshipping of unshipped units is pos-
sible and analyzing the effect of various parameters of intermodal interfaces on operational
decisions.
Appendices
Appendix A  Algorithms

Detailed pseudocodes of the algorithms used in this dissertation is presented in this section. Algorithm 1 shows the pseudocode for the Separate store and pick heuristic in Chapter 3. Pseudocodes for the Dual command store and pick and Multi-command store and pick heuristics in Chapter 3 are given in Algorithm 2 and Algorithm 3, respectively. The pseudocodes for the proposed heuristics in Chapter 4, Random location visit, Closest location visit, Shortest-time location visit, and Longest-time location visit heuristic are shown respectively in Algorithm 5, Algorithm 6, Algorithm 7, and Algorithm 8. Algorithm 9 shows the psuedocode for the route construction heuristic used in Chapter 5.
Algorithm 1 Multi-command operations heuristic 1: Separate store and pick

INPUT: $n_s, n_p, d_{ij}, s_{ij}$

1: $T_{\text{min}} \leftarrow M$  \Comment{M is a large number}
2: $R_r = \{0\}$ \Comment{Route}
3: $r \leftarrow 1$ \Comment{Trip number}
4: $\text{ToStore} = [2..n_s], \text{ToPick} = [n_s + 1..n_s + n_p + 1]$  
5: Generate a unique rank for each location \( g(i) = x \in [2, n_s + n_p + 1] \)
6: while $y \leq K$ do \Comment{$K =$ Maximum number of iterations}
7: while $i \leq n_s + n_p + 1$ do  \Comment{Store route generation}
8: \( g(i) = x \)
9: if $x \in \text{ToStore}$ then
10: \( R_r \leftarrow R_r, x \) \Comment{Add store location to trip}
11: delete $x$ from $\text{ToStore}$
12: while $j \leq n_s + n_p + 1$ do
13: Find closest location $j$
14: if $j \in \text{ToStore}$ then
15: \( R_r \leftarrow R_r, j \) \Comment{Add store location}
16: delete $j$ from $\text{ToStore}, r \leftarrow r + 1$ , Break While Loop
17: else
18: \( j \leftarrow j + 1 \)
19: \( R_r \leftarrow R_r, 1 \) \Comment{Return to depot}
20: $r \leftarrow r + 1, i \leftarrow i + 1$
21: else
22: \( i \leftarrow i + 1 \)
23: \( i \leftarrow 1 \)
24: while $i \leq n_s + n_p + 1$ do \Comment{Pick route generation}
25: \( g(i) = x \)
26: if $x \in \text{ToPick}$ then
27: $R_r \leftarrow R_r, x$ \Comment{Add pick location to trip}
28: delete $x$ from $\text{ToPick}$
29: while $j \leq n_s + n_p + 1$ do
30: Find closest location $j$
31: if $j \in \text{ToPick}$ then
32: \( R_r \leftarrow R_r, j \) \Comment{Add pick location}
33: delete $j$ from $\text{ToPick}, r \leftarrow r + 1$ , Break While Loop
34: else
35: \( j \leftarrow j + 1 \)
36: \( R_r \leftarrow R_r, 1 \) \Comment{Return to depot}
37: $r \leftarrow r + 1, i \leftarrow i + 1$
38: else
39: \( i \leftarrow i + 1 \)
40: Calculate Total time $T$ for Routes $\sum R_r \forall r$
41: if $T < T_{\text{min}}$ then
42: $T_{\text{min}} \leftarrow T$
43: BestRoute $\leftarrow R$
44: Return $T_{\text{min}}, \text{BestRoute}$
Algorithm 2 Multi-command operations heuristic 2: Store & pick with dual command

INPUT: \(n_s, n_p, d_{ij}, s_{ij}\)

1. \(T_{\text{min}} \leftarrow M\) \hspace{1cm} ▷ \(M\) is a large number
2. \(R_r = \{0\}\) \hspace{1cm} ▷ Route
3. \(r \leftarrow 1\) \hspace{1cm} ▷ Trip number
4. \(\text{ToStore} = [2..n_s], \text{ToPick} = [n_s + 1..n_s + n_p + 1]\)
5. Generate a unique rank for each location \(g(i) = x \in [2, n_s + n_p + 1]\)
6. while \(y \leq K\) do \hspace{1cm} ▷ \(K\) = Maximum number of iterations
7. \hspace{1cm} while \(i \leq n_s + n_p + 1\) do ▷ Dual command route generation
8. \hspace{2cm} \(g(i) = x\)
9. \hspace{2cm} if \(x \in \text{ToStore}\) then
10. \hspace{4cm} \(R_r \leftarrow R_r, x\) \hspace{1cm} ▷ Add store location to trip
11. \hspace{4cm} delete \(x\) from \(\text{ToStore}\)
12. \hspace{2cm} while \(j \leq n_s + n_p + 1\) do
13. \hspace{4cm} Find closest location \(j\)
14. \hspace{4cm} if \(j \in \text{ToPick}\) then
15. \hspace{6cm} \(R_r \leftarrow R_r, j\) \hspace{1cm} ▷ Add pick location
16. \hspace{6cm} delete \(j\) from \(\text{ToPick}\), \(r \leftarrow r + 1\) , Break While Loop
17. \hspace{4cm} else
18. \hspace{5cm} \(j \leftarrow j + 1\)
19. \hspace{4cm} \(R_r \leftarrow R_r, 1\) \hspace{1cm} ▷ Return to depot
20. \hspace{4cm} \(r \leftarrow r + 1, i \leftarrow i + 1\)
21. \hspace{2cm} else
22. \hspace{4cm} \(i \leftarrow i + 1\)
23. \hspace{2cm} \(i \leftarrow 1\)
24. \hspace{2cm} while \(i \leq n_s + n_p + 1\) do ▷ Pick route generation
25. \hspace{3cm} \(g(i) = x\)
26. \hspace{3cm} if \(x \in \text{ToPick}\) then
27. \hspace{4cm} \(R_r \leftarrow R_r, x\) \hspace{1cm} ▷ Add pick location to trip
28. \hspace{4cm} delete \(x\) from \(\text{ToPick}\)
29. \hspace{4cm} \(R_r \leftarrow R_r, 1\) \hspace{1cm} ▷ Return to depot
30. \hspace{3cm} else
31. \hspace{4cm} \(i \leftarrow i + 1\)
32. \hspace{2cm} Calculate Total time \(T\) for Routes \(\sum R_r\forall r\)
33. \hspace{2cm} if \(T < T_{\text{min}}\) then
34. \hspace{3cm} \(T_{\text{min}} \leftarrow T\)
35. \hspace{3cm} \(\text{BestRoute} \leftarrow R\)
36. \hspace{2cm} Return \(T_{\text{min}}, \text{BestRoute}\)
Algorithm 3: Multi-command operations heuristic 3: Multi-command operations

**INPUT:** $n_s, n_p, d_{ij}, s_{ij}\\n1: T_{\text{min}} \leftarrow M \quad \triangleright M$ is a large number\\n2: $R_r = \{0\} \quad \triangleright$ Route\\n3: $r \leftarrow 1 \quad \triangleright$ Trip number\\n4: $\text{ToStore} = [2..n_s], \text{ToPick} = [n_s + 1..n_s + n_p + 1]$\\n5: Generate a unique rank for each location $g(i) = x \in [2, n_s + n_p + 1]$\\n6: while $y \leq K$ do \quad $\triangleright K$ = Maximum number of iterations\\n7: \hspace{1em} while $i \leq n_s + n_p + 1$ do \quad $\triangleright$ Multi-command route generation\\n8: \hspace{2em} $g(i) = x$\\n9: \hspace{2em} if $x \in \text{ToStore}$ then\\n10: \hspace{3em} $R_r \leftarrow R_r, x$ \quad $\triangleright$ Add store location to trip\\n11: \hspace{3em} delete $x$ from $\text{ToStore}$\\n12: \hspace{2em} while $j \leq n_s + n_p + 1$ do\\n13: \hspace{3em} Find closest location $j$\\n14: \hspace{3em} if $j \in \text{ToStore}$ then\\n15: \hspace{4em} $R_r \leftarrow R_r, j$ \quad $\triangleright$ Add store location\\n16: \hspace{4em} delete $j$ from $\text{ToStore}$, Break While Loop\\n17: \hspace{3em} else\\n18: \hspace{4em} $j \leftarrow j + 1$\\n19: \hspace{2em} while $k \leq n_s + n_p + 1$ do\\n20: \hspace{3em} Find closest location $k$\\n21: \hspace{3em} if $k \in \text{ToPick}$ then\\n22: \hspace{4em} $R_r \leftarrow R_r, j$ \quad $\triangleright$ Add pick location\\n23: \hspace{4em} delete $k$ from $\text{ToPick}$, Break While Loop\\n24: \hspace{3em} else\\n25: \hspace{4em} $k \leftarrow k + 1$\\n26: \hspace{2em} while $k \leq n_s + n_p + 1$ do\\n27: \hspace{3em} Find closest location $l$\\n28: \hspace{3em} if $l \in \text{ToPick}$ then\\n29: \hspace{4em} $R_r \leftarrow R_r, j, l$ \quad $\triangleright$ Add pick location and return to depot\\n30: \hspace{4em} delete $l$ from $\text{ToPick}$, Break While Loop\\n31: \hspace{3em} else\\n32: \hspace{4em} $l \leftarrow l + 1$\\n33: \hspace{4em} $R_r \leftarrow R_r, 1$ \quad $\triangleright$ Return to depot\\n34: \hspace{2em} $r \leftarrow r + 1, i \leftarrow i + 1$\\n35: \hspace{2em} else\\n36: \hspace{3em} $i \leftarrow i + 1$\\n37: \hspace{3em} $i \leftarrow 1$
Algorithm 4 Multi-command operations heuristic 3: Multi-command operations

38: while $i \leq n_s + n_p + 1$ do \hspace{1cm} \triangleright \text{Pick route generation}
39: \hspace{1cm} g(i) = x
40: \hspace{1cm} if $x \in ToPick$ then
41: \hspace{1.5cm} $R_r \leftarrow R_r, x \hspace{1cm} \triangleright \text{Add pick location to trip}$
42: \hspace{1.5cm} delete $x$ from $ToPick$
43: \hspace{1cm} while $j \leq n_s + n_p + 1$ do
44: \hspace{1.5cm} find closest location $j$
45: \hspace{1.5cm} if $j \in ToPick$ then
46: \hspace{2cm} $R_r \leftarrow R_r, j \hspace{1cm} \triangleright \text{Add pick location}$
47: \hspace{2cm} delete $j$ from $ToPick$, $r \leftarrow r + 1$, Break While Loop
48: \hspace{1cm} else
49: \hspace{2cm} $j \leftarrow j + 1$
50: \hspace{1cm} $R_r \leftarrow R_r, 1 \hspace{1cm} \triangleright \text{Return to depot}$
51: \hspace{1cm} $r \leftarrow r + 1, i \leftarrow i + 1$
52: \hspace{1cm} else
53: \hspace{1.5cm} $i \leftarrow i + 1$
54: \hspace{1cm} Calculate Total time $T$ for Routes $\sum R_r \forall r$
55: \hspace{1cm} if $T < T_{\text{min}}$ then
56: \hspace{1.5cm} $T_{\text{min}} \leftarrow T$
57: \hspace{1.5cm} $BestRoute \leftarrow R$
58: \hspace{1cm} Return $T_{\text{min}}, BestRoute$
Algorithm 5 Random location visit route construction heuristic

**INPUT:** Distance matrix, store \((s_j)\) and pick \((p_j)\) list for each location, vehicle speed, unit handling times, vehicle capacity, maximum number of replications \((y_{max})\)

1: for \(y = 1\) To \(y_{max}\) do \(\triangleright\) Iterate for a pre-determined number of iterations
2: Generate a location visiting list \(L = L_1, L_2, ..., L_n\)
3: Read input data
4: \(TripNum = 1, i = 1\)
5: for \(i = 1\) to \(n\) do \(\triangleright\) Loop through location list
6: while \(s_{L_i} > 0\) or \(p_{L_i} > 0\) do \(\triangleright\) while demand at location \(L_i\) is not complete
7: \(\text{NumStore}(L_i) = \min(s_{L_i}, \text{VehicleCapacityLeft})\)
8: \(s_{L_i} \leftarrow s_{L_i} - \text{NumStore}(i)\)
9: \(\text{VehicleCapacityLeft} \leftarrow \text{VehicleCapacityLeft} - \text{NumStore}(L_i)\)
10: \(\text{NumPick}(L_i) = \min(p_{L_i}, \text{VehicleCapacityLeft})\)
11: \(p_{L_i} \leftarrow p_{L_i} - \text{NumPick}(L_i)\)
12: \(\text{VehicleCapacityLeft} \leftarrow \text{VehicleCapacityLeft} - \text{NumPick}(L_i)\)
13: if \(s_{L_i} > 0\) or \(p_{L_i} > 0\) then \(\triangleright\) Demand is not fulfilled
14: \(TripNum \leftarrow TripNum + 1\) \(\triangleright\) Return to the location in a new trip
15: \(\text{VehicleCapacityLeft} \leftarrow \text{VehicleCapacity}\)
16: else \(\triangleright\) Demand of location \(i\) is fulfilled
17: if \(\text{VehicleCapacityLeft} > 0\) then
18: if \(s_{L_{i+1}} > \text{VehicleCapacity}\) or \(p_{L_{i+1}} > \text{VehicleCapacity}\) or \(\text{VehicleCapacityLeft} > s_{L_{i+1}}\) or \(\text{VehicleCapacityLeft} > p_{L_{i+1}}\) then
19: \(L_i \leftarrow L_{i+1}\) \(\triangleright\) Visit next location on list in the same trip
20: else \(\triangleright\) Go back to depot and start a new trip
21: \(TripNum \leftarrow TripNum + 1\)
22: \(\text{VehicleCapacityLeft} \leftarrow \text{VehicleCapacity}\)
23: \(L_i \leftarrow L_{i+1}\)
24: for \(TripNum = 1\) to max. \(TripNum\) do
25: Rearrange visiting locations
26: Calculate route time for each trip
27: Select best visit order for each trip
28: \(\text{TotalRouteTime} = \sum_{TripNum} \text{RouteTime}\)
29: if \(\text{TotalRouteTime} < \text{BestRouteTime}\) then
30: \(\text{BestRouteTime} \leftarrow \text{TotalRouteTime}\)
31: Record Routes
Algorithm 6 Closest location visit route construction heuristic

**INPUT:** Distance matrix, store \((s_j)\) and pick \((p_j)\) list for each location, vehicle speed, unit handling times, vehicle capacity, maximum number of replications \((y_{max})\)

1: for \(y = 1\) To \(y_{max}\) do  \(\triangleright\) Iterate for a pre-determined number of iterations
2: Generate a random starting location \(L_1 < n\)
3: Read input data
4: \(\text{TripNum} = 1, i = L_1\)
5: for \(j = 1\) to \(n\) do  \(\triangleright\) Loop through location list
6: while \(s_i > 0\) or \(p_i > 0\) do  \(\triangleright\) Do while demand at location \(i\) is not complete
7: \(\text{NumStore}(L_i) = \min(s_i, \text{VehicleCapacityLeft})\)
8: \(s_i \leftarrow s_i - \text{NumStore}(L_i)\)
9: \(\text{VehicleCapacityLeft} \leftarrow \text{VehicleCapacityLeft} - \text{NumStore}(L_i)\)
10: \(\text{NumPick}(i) = \min(p_i, \text{VehicleCapacityLeft})\)
11: \(p_i \leftarrow p_i - \text{NumPick}(L_i)\)
12: \(\text{VehicleCapacityLeft} \leftarrow \text{VehicleCapacityLeft} - \text{NumPick}(L_i)\)
13: if \(s_{L_i} > 0\) or \(p_{L_i} > 0\) then  \(\triangleright\) Demand is not fulfilled
14: \(\text{TripNum} \leftarrow \text{TripNum} + 1\)  \(\triangleright\) Return to the location in a new trip
15: \(\text{VehicleCapacityLeft} \leftarrow \text{VehicleCapacity}\)
16: else  \(\triangleright\) Demand of location \(i\) is fulfilled
17: if \(\text{VehicleCapacityLeft} > 0\) then
18: Find the closest location \(L_{i+1}\) to current location \(L_i\) that has unfulfilled pick or store demand
19: if \(s_{L_{i+1}} > \text{VehicleCapacity}\) or \(p_{L_{i+1}} > \text{VehicleCapacity}\) or \(\text{VehicleCapacityLeft} > s_{L_{i+1}}\) or \(\text{VehicleCapacityLeft} > p_{L_{i+1}}\) then
20: \(L_i \leftarrow L_{i+1}\)  \(\triangleright\) Visit next location on list in the same trip
21: else  \(\triangleright\) Go back to depot and start a new trip
22: Find the closest location \(L_{i+1}\) to the depot that has unfulfilled pick or store demand
23: \(\text{TripNum} \leftarrow \text{TripNum} + 1\)
24: \(\text{VehicleCapacityLeft} \leftarrow \text{VehicleCapacity}\)
25: \(L_i \leftarrow L_{i+1}\)
26: for \(\text{TripNum} = 1\) to max. \(\text{TripNum}\) do
27: Rearrange visiting locations
28: Calculate route time for each trip
29: Select best visit order for each trip
30: \(\text{TotalRouteTime} = \sum_{\text{TripNum}} \text{Routetime}\)
31: if \(\text{TotalRouteTime} < \text{BestRouteTime}\) then
32: \(\text{BestRouteTime} \leftarrow \text{TotalRouteTime}\)
33: Record Routes
**Algorithm 7** Shortest-time location visit route construction heuristic

**INPUT:** Distance matrix, store \((s_j)\) and pick \((p_j)\) list for each location, vehicle speed, unit handling times, vehicle capacity, maximum number of replications \((y_{\text{max}})\)

1: for \(y = 1\) to \(y_{\text{max}}\) do  
   ▷ Iterate for a pre-determined number of iterations
2: Generate a random starting location \(L_1 < n\)
3: Read input data
4: \(\text{TripNum} = 1, i = L_1\)
5: for \(i = 1\) to \(n\) do  
   ▷ Loop through location list
6: while \(s_i > 0\) or \(p_i > 0\) do  
   ▷ while demand at location \(i\) is not complete
7: \(\text{NumStore}(i) = \min(s_i, \text{VehicleCapacityLeft})\)
8: \(s_i \leftarrow s_i - \text{NumStore}(i)\)
9: \(\text{VehicleCapacityLeft} \leftarrow \text{VehicleCapacityLeft} - \text{NumStore}(i)\)
10: \(\text{NumPick}(i) = \min(p_i, \text{VehicleCapacityLeft})\)
11: \(p_i \leftarrow p_i - \text{NumPick}(i)\)
12: \(\text{VehicleCapacityLeft} \leftarrow \text{VehicleCapacityLeft} - \text{NumPick}(i)\)
13: if \(s_i > 0\) or \(p_i > 0\) then  
   ▷ Demand is not fulfilled
14: \(\text{TripNum} \leftarrow \text{TripNum} + 1\)  
   ▷ Return to the location in a new trip
15: \(\text{VehicleCapacityLeft} \leftarrow \text{VehicleCapacity}\)
16: else  
   ▷ Demand of location \(i\) is fulfilled
17: if \(\text{VehicleCapacityLeft} > 0\) then
18: \(\text{Find the location } L_{i+1} \text{ with shortest route time from current location}\)
19: if \(s_{L+1} > \text{VehicleCapacity}\) or \(p_{L+1} > \text{VehicleCapacity}\) or \(\text{VehicleCapacityLeft} > s_{L+1}\) or \(\text{VehicleCapacityLeft} > p_{L+1}\) then
20: \(L_i \leftarrow L_{i+1}\)  
   ▷ Visit next location on list in the same trip
21: else  
   ▷ Go back to depot and start a new trip
22: \(\text{Find the location } L_{i+1} \text{ with shortest total route time from the depot}\)
23: \(\text{that has unfulfilled pick or store demand}\)
24: \(\text{TripNum} \leftarrow \text{TripNum} + 1\)
25: \(\text{VehicleCapacityLeft} \leftarrow \text{VehicleCapacity}\)
26: \(L_i \leftarrow L_{i+1}\)
27: for \(\text{TripNum} = 1\) to max. \(\text{TripNum}\) do
28: Rearrange visiting locations
29: Calculate route time for each trip
30: Select best visit order for each trip
31: \(\text{TotalRouteTime} = \sum_{\text{TripNum}} \text{Routetime}\)
32: if \(\text{TotalRouteTime} < \text{BestRouteTime}\) then
33: \(\text{BestRouteTime} \leftarrow \text{TotalRouteTime}\)
34: Record Routes
Algorithm 8 Longest-time location visit route construction heuristic

**INPUT:** Distance matrix, store \((s_j)\) and pick \((p_j)\) list for each location, vehicle speed, unit handling times, vehicle capacity, maximum number of replications \((y_{max})\)

1: for \(y = 1 \) to \(y_{max}\) do \(\triangleright\) Iterate for a pre-determined number of iterations
2: Generate a random starting location \(L_1 < n\)
3: Read input data
4: \(TripNum = 1, i = L_1\)
5: for \(i = 1\) to \(n\) do \(\triangleright\) Loop through location list
6: while \(s_i > 0\) or \(p_i > 0\) do \(\triangleright\) while demand at location \(i\) is not complete
7: \(\text{NumStore}(i) = \min.(s_i, \text{VehicleCapacityLeft})\)
8: \(s_i \leftarrow s_i - \text{NumStore}(i)\)
9: \(\text{VehicleCapacityLeft} \leftarrow \text{VehicleCapacityLeft} - \text{NumStore}(i)\)
10: \(\text{NumPick}(i) = \min.(p_i, \text{VehicleCapacityLeft})\)
11: \(p_i \leftarrow s_i - \text{NumPick}(i)\)
12: \(\text{VehicleCapacityLeft} \leftarrow \text{VehicleCapacityLeft} - \text{NumPick}(i)\)
13: if \(s_i > 0\) or \(p_i > 0\) then \(\triangleright\) Demand is not fulfilled
14: \(TripNum \leftarrow TripNum + 1\) \(\triangleright\) Return to the location in a new trip
15: \(\text{VehicleCapacityLeft} \leftarrow \text{VehicleCapacity}\)
16: else \(\triangleright\) Demand of location \(i\) is fulfilled
17: if \(\text{VehicleCapacityLeft} > 0\) then
18: Find the location \(L_{i+1}\) with shortest route time from current location \(L_i\) that has unfulfilled pick or store demand
19: if \(s_{L+1} > \text{VehicleCapacity}\) or \(p_{L+1} > \text{VehicleCapacity}\) or \(\text{VehicleCapacityLeft} > s_{L+1}\) or \(\text{VehicleCapacityLeft} > p_{L+1}\) then
20: \(L_i \leftarrow L_{i+1}\) \(\triangleright\) Visit next location on list in the same trip
21: else \(\triangleright\) Go back to depot and start a new trip
22: Find the location \(L_{i+1}\) with longest total route time from the depot that has unfulfilled pick or store demand
23: \(TripNum \leftarrow TripNum + 1\)
24: \(\text{VehicleCapacityLeft} \leftarrow \text{VehicleCapacity}\)
25: \(L_i \leftarrow L_{i+1}\)
26: for \(TripNum = 1\) to \(\text{max. TripNum}\) do
27: Rearrange visiting locations
28: Calculate route time for each trip
29: Select best visit order for each trip
30: \(\text{TotalRouteTime} = \sum_{\text{TripNum}} \text{Routetime}\)
31: if \(\text{TotalRouteTime} < \text{BestRouteTime}\) then
32: \(\text{BestRouteTime} \leftarrow \text{TotalRouteTime}\)
33: Record Routes

171
Algorithm 9 Construction heuristic for intermodal facility operations

**INPUT:** $p_{jk}, x, d_{lm}, s_f, s_b, t_s, t_p, T_{max}, y_{max}, n^t_{ij}, o^t_j$

1: for $t = 1$ To $T$ do ▶ For each time period
2: for $y = 1$ To $y_{max}$ do ▶ Iterate for a pre-determined number of iterations
3: Assign inbound dock doors randomly.
4: for $j = 1$ To $O$ do ▶ Direct outbound shipment assignments
5: if $O^t_j \geq 0$ then
6: for $i = 1$ to $L$ do
7: Find inbound dock door $l$ with largest $n^t_{ij}$
8: Find unassigned outbound dock door $m$ with minimum $d_{lm}$
9: Assign trailer to $j$ to outbound dock door $m$
10: for $i = 1$ to $I$ do
11: $Shipped^t_{ij} = \max. (n^t_{ij}, CapacityLeft^t_j)$.
12: if $Shipped^t_{ij} > 0$ and $ElapsedTime < v \times T_{max}$ then
13: Generate internal routes. Calculate Total route time and Cost.
14: $ElapsedTime \leftarrow ElapsedTime +$ route time
15: $Unshipped^t_{ij} = \max. (n^t_{ij} - CapacityLeft^t_j, 0)$.
16: $CapacityLeft^t_j = O^t_j - Shipped^t_{ij}$
for \( i = 1 \) To \( I \) do
  for \( j = 1 \) To \( O \) do
    if \( Unshipped_{ij} > 0 \) then
      Find alternate locations \( k \) with \( CapacityLeft_{tk} > 0 \)
      Choose location \( k \) with minimum alternate shipping cost
      if Alternate Shipping Cost < Storage Cost then
        Reship_{jk} = \text{min.} (Unshipped_{ij}, CapacityLeft_{tk})
        if ReShip_{jk} > 0 and ElapsedTime < \( v \times T_{max} \) then
          Generate routes. Calculate Total route time and Cost.
          ElapsedTime \leftarrow ElapsedTime + \text{route time}
        Unshipped_{ij} \leftarrow Unshipped_{ij} - \text{Reship}_{jk}
        CapacityLeft_{tk} \leftarrow CapacityLeft_{tk} - \text{Reship}_{jk}
    if \( Unshipped_{ij} > 0 \) then
      if Storage Cost < Alternate Shipping Cost then
        Stored_{ij} \leftarrow \min.(Unshipped_{ij}, S - UsedStorage).
        Unshipped_{ij} \leftarrow Unshipped_{ij} - Stored_{ij}
        UsedStorage \leftarrow UsedStorage + Stored_{ij}
      else
        Reship_{jk} = \text{min.} (Unshipped_{ij}, CapacityLeft_{tk})
        if ReShip_{jk} > 0 and ElapsedTime < \( v \times T_{max} \) then
          Generate routes. Calculate Total route time and Cost.
          ElapsedTime \leftarrow ElapsedTime + \text{route time}
        Unshipped_{ij} \leftarrow Unshipped_{ij} - \text{Reship}_{jk}
        CapacityLeft_{tk} \leftarrow CapacityLeft_{tk} - \text{Reship}_{jk}
      if \( Unshipped_{ij} > 0 \) then
        Create an extra outbound shipment to \( j \).
        Unshipped_{ij} \leftarrow 0
    Calculate Total Cost for the iteration \( y \)
  Choose BestSolution with lowest Total Cost for time period \( t \)
  for \( i = 1 \) To \( I \) do
    for \( j = 1 \) To \( O \) do
      \( n_{ij} \leftarrow n_{ij} + Stored_{ij} \)
      \( Stored_{ij} \leftarrow 0 \)
      \( UsedStorage \leftarrow 0 \)
### Appendix B  Sample problems and output data

#### B.1 Multi-command operations

| Location | 0  | 1  | 2  | 3  | 4  | 5  | 6  | 7  | 8  | 9  | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| 0        | 0  | 95 | 60 | 150| 70 | 235| 185| 265| 200| 20 | 150 | 305| 190| 230| 105| 315| 195| 90 | 35 | 250| 215|
| 1        | 95 | 0  | 130| 190|105| 250| 225| 300| 245| 105| 185 | 350| 225| 265| 140| 350| 230| 3 | 90 | 290| 260|
| 2        | 60 | 130| 0  | 180|105| 265| 215| 305| 230| 75 | 185 | 335| 230| 270| 140| 350| 230| 125| 70 | 280| 245|
| 3        | 150| 190| 130| 0  |105| 115| 125| 155| 80 | 165| 90  | 185| 80 | 120| 80 | 200| 80 | 185| 150| 130| 115|
| 4        | 70 | 105| 105| 105|0  | 195| 140| 215| 160| 80 | 100 | 265| 140| 150| 55 | 265| 140| 100| 65 | 205| 175|
| 5        | 235| 280| 280| 115|195| 0  | 210| 135| 125| 250| 185| 170| 90 | 140| 170| 185| 110| 275| 235| 115| 200|
| 6        | 185| 225| 215| 125|140| 210| 0  | 250| 175| 200| 130| 280| 175| 215| 115| 295| 175| 225| 185| 225| 190|
| 7        | 265| 300| 305| 195|215| 135| 250| 0  | 160| 275| 205| 205| 115| 35 | 190| 210| 130| 295| 260| 150| 235|
| 8        | 200| 245| 230| 80 |160| 125| 175| 160| 0  | 215| 150| 195| 85 | 125| 135| 210| 90 | 240| 200| 140| 165|
| 9        | 20 | 105| 75 |165| 80 | 250| 200| 275| 215| 0  | 160| 320| 200| 240| 115| 325| 205| 100| 45 | 265| 230|
| 10       | 150| 185| 185| 95 |100| 185| 130| 205| 150| 100| 0  | 255| 130| 170| 75  | 255| 135| 180| 145| 195| 165|
| 11       | 305| 350| 350| 185|265| 170| 260| 205| 195| 320| 255| 0  | 160| 170| 240| 145| 180| 345| 305| 355| 270|
| 12       | 190| 225| 230| 80 |140| 90 | 135| 115| 85 | 200| 130| 160| 0  | 80  | 115| 165| 15 | 220| 185| 185| 165|
| 13       | 230| 265| 270| 120|180| 100| 215| 35 | 125| 240| 170| 170| 80  | 155| 175| 95 | 260| 225| 115| 200|
| 14       | 105| 140| 140| 80 |55 | 170| 115| 190| 135| 115| 75 | 240| 115| 155| 0  | 240| 120| 135| 100| 180| 150|
| 15       | 315| 350| 350| 200|265| 185| 295| 210| 210| 325| 255| 145| 165| 175| 240| 0  | 180| 345| 310| 90 | 285|
| 16       | 195| 230| 230| 80 |145| 110| 175| 130| 90 | 205| 135| 180| 15 | 95  | 120| 180 | 0  | 225| 190| 120| 165|
| 17       | 90 | 5 | 125| 185|100| 275| 220| 285| 240| 100| 180| 345| 220| 260| 135| 345| 225| 0  | 85 | 285| 235|
| 18       | 35 | 90 | 70 |130 |65 | 235| 185| 260| 200| 45 | 145| 305| 185| 225| 180| 310| 190| 85 | 0  | 250| 215|
| 19       | 250| 290| 280| 130|205| 115| 225| 150| 140| 265| 195| 55 | 105| 115| 180| 90 | 120| 285| 250| 0  | 215|

Table 1: Distance matrix for a 20 location problem. All distances are in feet. Location 0 is the depot. 1 is stackable, 0 is not-stackable.
Table 2: Stackability matrix for a 20 pallet problem with 50% stackability density.

<table>
<thead>
<tr>
<th>Pallet</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>17</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 3: Sample set of routes generated by the MILP for a 20 pallet problem

<table>
<thead>
<tr>
<th>x02</th>
<th>5</th>
<th>6</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td>x04</td>
<td>8</td>
<td>7</td>
<td>17</td>
</tr>
<tr>
<td>x04</td>
<td>10</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>x07</td>
<td>2</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>x07</td>
<td>3</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>x07</td>
<td>11</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>x09</td>
<td>9</td>
<td>15</td>
<td>19</td>
</tr>
</tbody>
</table>

Objective 25.37
B.2 Generalized model for multi-command operations

Table 4: Distance matrix and store & pick list for a sample 20 location problem. Distances are in feet.

<table>
<thead>
<tr>
<th>Location</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>180</td>
<td>195</td>
<td>160</td>
<td>135</td>
<td>100</td>
<td>45</td>
<td>90</td>
<td>100</td>
<td>150</td>
<td>160</td>
<td>205</td>
<td>210</td>
<td>160</td>
<td>195</td>
<td>150</td>
<td>170</td>
<td>195</td>
<td>170</td>
<td>170</td>
<td>170</td>
</tr>
<tr>
<td>1</td>
<td>180</td>
<td>195</td>
<td>185</td>
<td>150</td>
<td>125</td>
<td>100</td>
<td>60</td>
<td>120</td>
<td>135</td>
<td>185</td>
<td>200</td>
<td>245</td>
<td>250</td>
<td>200</td>
<td>250</td>
<td>215</td>
<td>235</td>
<td>220</td>
<td>230</td>
<td>240</td>
<td>240</td>
</tr>
<tr>
<td>2</td>
<td>195</td>
<td>185</td>
<td>185</td>
<td>155</td>
<td>130</td>
<td>105</td>
<td>65</td>
<td>120</td>
<td>135</td>
<td>185</td>
<td>205</td>
<td>245</td>
<td>255</td>
<td>205</td>
<td>255</td>
<td>215</td>
<td>235</td>
<td>210</td>
<td>220</td>
<td>225</td>
<td>230</td>
</tr>
<tr>
<td>3</td>
<td>15</td>
<td>185</td>
<td>190</td>
<td>165</td>
<td>135</td>
<td>110</td>
<td>70</td>
<td>130</td>
<td>145</td>
<td>195</td>
<td>210</td>
<td>250</td>
<td>260</td>
<td>210</td>
<td>260</td>
<td>230</td>
<td>245</td>
<td>215</td>
<td>225</td>
<td>235</td>
<td>240</td>
</tr>
<tr>
<td>4</td>
<td>205</td>
<td>200</td>
<td>190</td>
<td>160</td>
<td>135</td>
<td>110</td>
<td>70</td>
<td>130</td>
<td>145</td>
<td>195</td>
<td>210</td>
<td>250</td>
<td>260</td>
<td>210</td>
<td>260</td>
<td>230</td>
<td>245</td>
<td>215</td>
<td>225</td>
<td>235</td>
<td>240</td>
</tr>
<tr>
<td>5</td>
<td>205</td>
<td>200</td>
<td>190</td>
<td>160</td>
<td>135</td>
<td>110</td>
<td>70</td>
<td>130</td>
<td>145</td>
<td>195</td>
<td>210</td>
<td>250</td>
<td>260</td>
<td>210</td>
<td>260</td>
<td>230</td>
<td>245</td>
<td>215</td>
<td>225</td>
<td>235</td>
<td>240</td>
</tr>
<tr>
<td>6</td>
<td>15</td>
<td>185</td>
<td>190</td>
<td>165</td>
<td>135</td>
<td>110</td>
<td>70</td>
<td>130</td>
<td>145</td>
<td>195</td>
<td>210</td>
<td>250</td>
<td>260</td>
<td>210</td>
<td>260</td>
<td>230</td>
<td>245</td>
<td>215</td>
<td>225</td>
<td>235</td>
<td>240</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
<td>125</td>
<td>135</td>
<td>135</td>
<td>115</td>
<td>100</td>
<td>65</td>
<td>120</td>
<td>135</td>
<td>185</td>
<td>200</td>
<td>245</td>
<td>255</td>
<td>205</td>
<td>255</td>
<td>215</td>
<td>235</td>
<td>210</td>
<td>220</td>
<td>225</td>
<td>230</td>
</tr>
<tr>
<td>8</td>
<td>150</td>
<td>210</td>
<td>220</td>
<td>200</td>
<td>140</td>
<td>110</td>
<td>70</td>
<td>130</td>
<td>145</td>
<td>195</td>
<td>210</td>
<td>250</td>
<td>260</td>
<td>210</td>
<td>260</td>
<td>230</td>
<td>245</td>
<td>215</td>
<td>225</td>
<td>235</td>
<td>240</td>
</tr>
<tr>
<td>9</td>
<td>160</td>
<td>215</td>
<td>225</td>
<td>205</td>
<td>140</td>
<td>115</td>
<td>70</td>
<td>130</td>
<td>145</td>
<td>195</td>
<td>210</td>
<td>250</td>
<td>260</td>
<td>210</td>
<td>260</td>
<td>230</td>
<td>245</td>
<td>215</td>
<td>225</td>
<td>235</td>
<td>240</td>
</tr>
<tr>
<td>10</td>
<td>145</td>
<td>200</td>
<td>205</td>
<td>160</td>
<td>135</td>
<td>110</td>
<td>70</td>
<td>130</td>
<td>145</td>
<td>195</td>
<td>210</td>
<td>250</td>
<td>260</td>
<td>210</td>
<td>260</td>
<td>230</td>
<td>245</td>
<td>215</td>
<td>225</td>
<td>235</td>
<td>240</td>
</tr>
<tr>
<td>12</td>
<td>150</td>
<td>200</td>
<td>205</td>
<td>160</td>
<td>135</td>
<td>110</td>
<td>70</td>
<td>130</td>
<td>145</td>
<td>195</td>
<td>210</td>
<td>250</td>
<td>260</td>
<td>210</td>
<td>260</td>
<td>230</td>
<td>245</td>
<td>215</td>
<td>225</td>
<td>235</td>
<td>240</td>
</tr>
<tr>
<td>13</td>
<td>180</td>
<td>220</td>
<td>225</td>
<td>215</td>
<td>150</td>
<td>125</td>
<td>80</td>
<td>140</td>
<td>155</td>
<td>205</td>
<td>225</td>
<td>265</td>
<td>275</td>
<td>230</td>
<td>275</td>
<td>245</td>
<td>265</td>
<td>235</td>
<td>250</td>
<td>255</td>
<td>260</td>
</tr>
<tr>
<td>14</td>
<td>260</td>
<td>225</td>
<td>225</td>
<td>215</td>
<td>150</td>
<td>125</td>
<td>80</td>
<td>140</td>
<td>155</td>
<td>205</td>
<td>225</td>
<td>265</td>
<td>275</td>
<td>230</td>
<td>275</td>
<td>245</td>
<td>265</td>
<td>235</td>
<td>250</td>
<td>255</td>
<td>260</td>
</tr>
<tr>
<td>15</td>
<td>160</td>
<td>200</td>
<td>205</td>
<td>160</td>
<td>135</td>
<td>110</td>
<td>70</td>
<td>130</td>
<td>145</td>
<td>195</td>
<td>210</td>
<td>250</td>
<td>260</td>
<td>210</td>
<td>260</td>
<td>230</td>
<td>245</td>
<td>215</td>
<td>225</td>
<td>235</td>
<td>240</td>
</tr>
<tr>
<td>16</td>
<td>150</td>
<td>200</td>
<td>205</td>
<td>160</td>
<td>135</td>
<td>110</td>
<td>70</td>
<td>130</td>
<td>145</td>
<td>195</td>
<td>210</td>
<td>250</td>
<td>260</td>
<td>210</td>
<td>260</td>
<td>230</td>
<td>245</td>
<td>215</td>
<td>225</td>
<td>235</td>
<td>240</td>
</tr>
<tr>
<td>17</td>
<td>180</td>
<td>210</td>
<td>215</td>
<td>205</td>
<td>160</td>
<td>135</td>
<td>110</td>
<td>70</td>
<td>130</td>
<td>145</td>
<td>195</td>
<td>210</td>
<td>250</td>
<td>260</td>
<td>210</td>
<td>260</td>
<td>230</td>
<td>245</td>
<td>215</td>
<td>225</td>
<td>230</td>
</tr>
<tr>
<td>Store</td>
<td>0</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>7</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>0</td>
<td>6</td>
<td>3</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>Pick</td>
<td>0</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>0</td>
<td>3</td>
<td>7</td>
<td>6</td>
<td>1</td>
<td>7</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>0</td>
<td>6</td>
<td>3</td>
<td>5</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 5: Sample set of routes generated by Shortest-time location visit algorithm for a 20 location problem. Each trip starts and ends at the depot.
Table 6: Total route time vs. unit load pick time for different route construction methods. Times are in minutes. Gap is the difference between the maximum and minimum value as a percentage of the minimum.

<table>
<thead>
<tr>
<th>Pick time</th>
<th>Random</th>
<th>Closest</th>
<th>Shortest</th>
<th>Longest</th>
<th>Gap (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Travel</td>
<td>Handling</td>
<td>Travel</td>
<td>Handling</td>
<td>Travel</td>
</tr>
<tr>
<td>0.1</td>
<td>38.00</td>
<td>50.37</td>
<td>32.30</td>
<td>50.42</td>
<td>33.88</td>
</tr>
<tr>
<td>0.2</td>
<td>38.12</td>
<td>63.38</td>
<td>32.30</td>
<td>63.46</td>
<td>32.80</td>
</tr>
<tr>
<td>0.3</td>
<td>38.00</td>
<td>76.38</td>
<td>32.33</td>
<td>76.44</td>
<td>32.06</td>
</tr>
<tr>
<td>0.4</td>
<td>38.08</td>
<td>89.39</td>
<td>32.30</td>
<td>89.47</td>
<td>31.83</td>
</tr>
<tr>
<td>0.5</td>
<td>38.05</td>
<td>102.39</td>
<td>32.35</td>
<td>102.51</td>
<td>32.33</td>
</tr>
<tr>
<td>0.6</td>
<td>38.10</td>
<td>115.39</td>
<td>32.31</td>
<td>115.52</td>
<td>32.92</td>
</tr>
<tr>
<td>0.7</td>
<td>38.05</td>
<td>128.39</td>
<td>32.34</td>
<td>128.56</td>
<td>33.01</td>
</tr>
<tr>
<td>0.8</td>
<td>38.03</td>
<td>141.39</td>
<td>32.37</td>
<td>141.55</td>
<td>33.12</td>
</tr>
<tr>
<td>0.9</td>
<td>38.00</td>
<td>154.38</td>
<td>32.30</td>
<td>154.56</td>
<td>33.15</td>
</tr>
<tr>
<td>1.0</td>
<td>37.97</td>
<td>167.39</td>
<td>32.30</td>
<td>167.59</td>
<td>33.38</td>
</tr>
</tbody>
</table>

Gap (%) | 0.37 | 232.31 | 0.23 | 232.42 | 6.44 | 232.57 | 11.72 | 234.07 | 0.00 | 0.00 |

Table 7: Total route time vs. unit load store time variation for different route construction heuristics. Times are in minutes. Gap is the difference between the maximum and minimum value as a percentage of the minimum.

<table>
<thead>
<tr>
<th>Store time</th>
<th>Random</th>
<th>Closest</th>
<th>Shortest</th>
<th>Longest</th>
<th>Gap (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Travel</td>
<td>Handling</td>
<td>Travel</td>
<td>Handling</td>
<td>Travel</td>
</tr>
<tr>
<td>0.1</td>
<td>37.97</td>
<td>51.46</td>
<td>32.30</td>
<td>51.52</td>
<td>31.43</td>
</tr>
<tr>
<td>0.2</td>
<td>38.06</td>
<td>63.92</td>
<td>32.30</td>
<td>63.99</td>
<td>31.68</td>
</tr>
<tr>
<td>0.3</td>
<td>37.97</td>
<td>76.38</td>
<td>32.30</td>
<td>76.49</td>
<td>32.08</td>
</tr>
<tr>
<td>0.4</td>
<td>38.03</td>
<td>88.85</td>
<td>32.30</td>
<td>88.91</td>
<td>32.80</td>
</tr>
<tr>
<td>0.5</td>
<td>38.00</td>
<td>101.30</td>
<td>32.40</td>
<td>101.37</td>
<td>33.40</td>
</tr>
<tr>
<td>0.6</td>
<td>37.97</td>
<td>113.74</td>
<td>32.30</td>
<td>113.82</td>
<td>34.15</td>
</tr>
<tr>
<td>0.7</td>
<td>38.17</td>
<td>126.23</td>
<td>32.34</td>
<td>126.35</td>
<td>34.27</td>
</tr>
<tr>
<td>0.8</td>
<td>37.97</td>
<td>138.65</td>
<td>32.34</td>
<td>138.83</td>
<td>34.49</td>
</tr>
<tr>
<td>0.9</td>
<td>38.15</td>
<td>151.11</td>
<td>32.30</td>
<td>151.19</td>
<td>34.73</td>
</tr>
<tr>
<td>1.0</td>
<td>38.10</td>
<td>163.58</td>
<td>32.32</td>
<td>163.65</td>
<td>34.74</td>
</tr>
</tbody>
</table>

Gap (%) | 0.51 | 217.86 | 0.31 | 217.63 | 10.53 | 218.05 | 4.41  | 215.94 | 0.00 | 0.00 |

Table 8: Total route time vs. unit load store and pick time for different route construction heuristics. Times are in minutes. Gap is the difference between the maximum and minimum value as a percentage of the minimum.
Table 9: Total route time vs. vehicle speed for different route construction heuristics. Speed is in feet/min. Times are in minutes. Gap is the difference between the maximum and minimum value as a percentage of the minimum.

<table>
<thead>
<tr>
<th>Value</th>
<th>200</th>
<th>300</th>
<th>400</th>
<th>500</th>
<th>600</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>133.34</td>
<td>114.40</td>
<td>105.01</td>
<td>99.24</td>
<td>95.36</td>
</tr>
<tr>
<td>Random Min</td>
<td>119.77</td>
<td>103.37</td>
<td>94.90</td>
<td>89.93</td>
<td>86.62</td>
</tr>
<tr>
<td>Random Max</td>
<td>147.80</td>
<td>124.45</td>
<td>112.77</td>
<td>105.77</td>
<td>101.10</td>
</tr>
<tr>
<td>Closest</td>
<td>124.91</td>
<td>108.76</td>
<td>100.67</td>
<td>95.85</td>
<td>92.60</td>
</tr>
<tr>
<td>Closest Min</td>
<td>112.42</td>
<td>98.25</td>
<td>91.17</td>
<td>86.92</td>
<td>84.09</td>
</tr>
<tr>
<td>Closest Max</td>
<td>135.99</td>
<td>116.88</td>
<td>107.70</td>
<td>102.20</td>
<td>98.53</td>
</tr>
<tr>
<td>Shortest</td>
<td>124.88</td>
<td>108.65</td>
<td>101.12</td>
<td>96.15</td>
<td>92.99</td>
</tr>
<tr>
<td>Shortest Min</td>
<td>113.25</td>
<td>99.55</td>
<td>93.30</td>
<td>88.40</td>
<td>85.20</td>
</tr>
<tr>
<td>Shortest Max</td>
<td>135.81</td>
<td>118.01</td>
<td>108.59</td>
<td>103.45</td>
<td>99.61</td>
</tr>
<tr>
<td>Longest</td>
<td>121.72</td>
<td>106.85</td>
<td>99.93</td>
<td>95.21</td>
<td>91.93</td>
</tr>
<tr>
<td>Longest Min</td>
<td>111.32</td>
<td>98.68</td>
<td>91.50</td>
<td>87.75</td>
<td>84.67</td>
</tr>
<tr>
<td>Longest Max</td>
<td>133.31</td>
<td>112.14</td>
<td>104.74</td>
<td>101.09</td>
<td>97.49</td>
</tr>
</tbody>
</table>

Table 10: Total route time vs. vehicle capacity for different route construction heuristics. Times are in minutes. Gap is the difference between the maximum and minimum value as a percentage of the minimum.

<table>
<thead>
<tr>
<th>Vehicle capacity</th>
<th>200</th>
<th>300</th>
<th>400</th>
<th>500</th>
<th>600</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>65.51</td>
<td>80.79</td>
<td>58.77</td>
<td>80.87</td>
<td>56.83</td>
</tr>
<tr>
<td>Closest</td>
<td>48.07</td>
<td>78.18</td>
<td>41.07</td>
<td>78.17</td>
<td>40.55</td>
</tr>
<tr>
<td>Shortest</td>
<td>37.97</td>
<td>76.38</td>
<td>32.31</td>
<td>76.47</td>
<td>32.06</td>
</tr>
<tr>
<td>Longest</td>
<td>32.27</td>
<td>75.19</td>
<td>26.24</td>
<td>75.17</td>
<td>27.20</td>
</tr>
<tr>
<td>Gap (%)</td>
<td>145.71</td>
<td>9.05</td>
<td>158.99</td>
<td>9.20</td>
<td>141.83</td>
</tr>
</tbody>
</table>

Table 11: Total route time vs. vehicle capacity for different route construction heuristics. Times are in minutes. Gap is the difference between the maximum and minimum value as a percentage of the minimum.
Table 12: Total route time vs. vehicle capacity for different route construction heuristics—minimum & maximum values. Times are in minutes.

<table>
<thead>
<tr>
<th>Number of locations</th>
<th>Random Travel Handling</th>
<th>Closest Travel Handling</th>
<th>Shortest Travel Handling</th>
<th>Longest Travel Handling</th>
<th>Gap (%) Travel Handling</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>38.03 76.37</td>
<td>32.32 76.44</td>
<td>32.10 76.55</td>
<td>30.61 76.24</td>
<td>24.25 0.41</td>
</tr>
<tr>
<td>50</td>
<td>96.91 189.94</td>
<td>77.00 190.22</td>
<td>80.81 190.54</td>
<td>76.95 189.87</td>
<td>25.94 0.35</td>
</tr>
<tr>
<td>100</td>
<td>232.01 365.49</td>
<td>161.44 366.22</td>
<td>169.54 366.87</td>
<td>176.19 365.82</td>
<td>43.71 0.38</td>
</tr>
<tr>
<td>200</td>
<td>467.71 736.13</td>
<td>316.31 737.44</td>
<td>331.97 738.98</td>
<td>347.28 736.89</td>
<td>47.87 0.29</td>
</tr>
</tbody>
</table>

Gap (%) 1,129.80 863.86 878.83 864.71 931.32 865.34 1,034.59 866.57 0.00 0.00

Table 13: Total route time vs. number of locations for different route construction heuristics. Times are in minutes. Gap is the difference between the maximum and minimum value as a percentage of the minimum.

<table>
<thead>
<tr>
<th>Value</th>
<th>Number of locations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>20    50    100    200</td>
</tr>
<tr>
<td>Random Min</td>
<td>114.40 286.85 597.50 1,203.84</td>
</tr>
<tr>
<td>Random Max</td>
<td>103.37 261.16 554.38 1,117.16</td>
</tr>
<tr>
<td>Closest</td>
<td>124.45 304.25 646.61 1,303.97</td>
</tr>
<tr>
<td>Closest Min</td>
<td>98.25 244.28 491.42 980.98</td>
</tr>
<tr>
<td>Closest Max</td>
<td>118.01 284.55 582.79 1,125.75</td>
</tr>
<tr>
<td>Shortest</td>
<td>116.88 277.77 570.42 1,106.71</td>
</tr>
<tr>
<td>Shortest Min</td>
<td>108.65 271.35 536.41 1,070.05</td>
</tr>
<tr>
<td>Shortest Max</td>
<td>99.55 249.18 501.76 998.56</td>
</tr>
<tr>
<td>Longest</td>
<td>114.12 278.11 580.14 1,141.37</td>
</tr>
</tbody>
</table>

Table 14: Total route time vs. number of locations for different route construction heuristics—minimum & maximum values. Times are in minutes.
Table 15: Total route time vs. input range for different route construction heuristics. Times are in minutes. Gap is the difference between the maximum and minimum value as a percentage of the minimum.
### B.3 Intermodal interface operations

<table>
<thead>
<tr>
<th>Location</th>
<th>InboundQuantity</th>
<th>OutboundQuantity</th>
<th>Destination1</th>
<th>Quantity 1</th>
<th>Destination 2</th>
<th>Quantity 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>15</td>
<td>0</td>
<td>11</td>
<td>15</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>15</td>
<td>15</td>
<td>0</td>
<td>8</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>25</td>
<td>20</td>
<td>10</td>
<td>5</td>
<td>15</td>
<td>20</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
<td>10</td>
<td>2</td>
<td>10</td>
<td>11</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>20</td>
<td>6</td>
<td>5</td>
<td>13</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>20</td>
<td>5</td>
<td>14</td>
<td>0</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>7</td>
<td>25</td>
<td>15</td>
<td>18</td>
<td>10</td>
<td>18</td>
<td>15</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>10</td>
<td>13</td>
<td>5</td>
<td>16</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>35</td>
<td>1</td>
<td>0</td>
<td>17</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>25</td>
<td>5</td>
<td>5</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>11</td>
<td>15</td>
<td>10</td>
<td>3</td>
<td>15</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>15</td>
<td>15</td>
<td>17</td>
<td>5</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td>13</td>
<td>20</td>
<td>15</td>
<td>7</td>
<td>20</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>10</td>
<td>25</td>
<td>9</td>
<td>5</td>
<td>19</td>
<td>5</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>30</td>
<td>16</td>
<td>0</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>16</td>
<td>15</td>
<td>25</td>
<td>4</td>
<td>10</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>17</td>
<td>10</td>
<td>5</td>
<td>8</td>
<td>10</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>15</td>
<td>30</td>
<td>19</td>
<td>5</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>19</td>
<td>25</td>
<td>5</td>
<td>12</td>
<td>0</td>
<td>14</td>
<td>25</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
<td>5</td>
<td>20</td>
<td>0</td>
<td>3</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 16: Sample inbound & outbound schedule with an outbound/inbound capacity ratio of 1.25 for one time period in an intermodal facility.
<table>
<thead>
<tr>
<th>Location</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>4</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>7</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>13</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>0</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>15</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>18</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>0</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>19</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 17: Sample reship cost matrix of an intermodal facility for shipping to alternate destinations. Reship cost is uniformly distributed [1,5].
Bibliography


