University of Central Florida
STARS

Electronic Theses and Dissertations, 2004-2019

2015

Modeling Dense Storage Systems With Location Uncertainty

Mohamed Awwad
University of Central Florida

Part of the Industrial Engineering Commons
Find similar works at: https://stars.library.ucf.edu/etd
University of Central Florida Libraries http://library.ucf.edu

This Doctoral Dissertation (Open Access) is brought to you for free and open access by STARS. It has been accepted for inclusion in Electronic Theses and Dissertations, 2004-2019 by an authorized administrator of STARS. For more information, please contact STARS@ucf.edu.

STARS Citation
Awwad, Mohamed, "Modeling Dense Storage Systems With Location Uncertainty" (2015). *Electronic Theses and Dissertations, 2004-2019*. 5148.
https://stars.library.ucf.edu/etd/5148

MODELING DENSE STORAGE SYSTEMS WITH LOCATION UNCERTAINTY

by

MOHAMED A. AWWAD
B.S. Mechanical Design and Production Engineering, Cairo University, 2009
M.S. Mechanical Design and Production Engineering, Cairo University, 2013
M.S. Industrial Engineering, University of Central Florida, 2014

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in the Department of Industrial Engineering and Management Systems
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Fall Term
2015

Major Professor:
Jennifer A. Pazour

© 2015 Mohamed A. Awwad

ABSTRACT

This dissertation focuses on developing models to study the problem of searching and retrieving items in a dense storage environment. We consider a special storage configuration called an inverted T configuration, which has one horizontal and one vertical aisle. Inverted T configurations have fewer aisles than a traditional aisle-based storage environment. This increases the storage density; however, requires that some items to be moved out of the way to gain access to other more deeply stored items. Such movement can result in item location uncertainty. When items are requested for retrieval in a dense storage environment with item location uncertainty, searching is required.

Dense storage has a practical importance as it allows for the use of available space efficiently, which is especially important with the scarce and expensive space onboard of US Navy's ships that form a sea base. A sea base acts as a floating distribution center that provides ready issue material to forces ashore participating in various types of missions. The sea basing concept and the importance of a sea base's responsiveness is our main motivation to conduct this research.

In chapter 2, we review three major bodies of literature: 1) sea based logistics, 2) dense storage and 3) search theory. Sea based logistics literature mostly focuses on the concept and the architecture of a sea base, with few papers developing mathematical models to solve operational problems of a sea base, including papers handling the logistical and sustainment aspects. Literature related to dense storage can be broken down into work dealing with a dense storage environment with an inverted T configuration and other papers dealing with other dense storage configurations. It was found that some of the dense storage literature was motivated by the same application, i.e. sea based logistics. Finally, we surveyed the vast search theory literature and classification of search environments. This research contributes to the

intersection of these three bodies of literature. Specifically, this research, motivated by the application of sea basing, develops search heuristics for dense storage environments that require moving items out of the way during searching.

In chapter 3, we present the problem statements. We study two single-searcher search problems. The first problem is searching for a single item in an inverted T dense storage environment. The second one is searching for one or more items in an inverted T storage environment with items stacked over each other in the vertical direction.

In chapter 4, we present our first contribution. In this contribution we propose a search plan heuristic to search for a single item in an inverted T, *k*-deep dense storage system with the objective of decreasing the expected search time in such an environment. In this contribution, we define each storage environment entirely by the accessibility constant and the storeroom length. In addition, equations are derived to calculate each component of the search time equation that we propose: travel, put-back and repositioning. Two repositioning policies are studied. We find that a repositioning policy that uses the open aisle locations as temporary storage locations and requires put-back of these items while searching is recommended. This recommendation is because such a policy results in lower expected search time and lower variability than a policy that uses available space outside the storage area and handles put-back independently of the search process.

Statistical analysis is used to analyze the numerical results of the first contribution and to analyze the performances of both repositioning polices. We derive the probability distribution of search times in a storeroom with small configurations in terms of the accessibility constant and length. It was found that this distribution can be approximated using a lognormal probability distribution with a certain mean and standard deviation. Knowing the probability

distribution provides the decision makers with the full range of all possible probabilities of search times, which is useful for downstream planning operations.

In chapter 5, we present the second contribution, in which we propose a search plan heuristic but for multiple items in an inverted T, k-deep storage system. Additionally, we consider stacking multiple items over each other. Stacking items over each other, increases the number of stored items and allows for the utilization of the vertical space. In this second contribution, we are using the repositioning policy that proved its superiority in the first contribution. This contribution investigates a more general and a much more challenging environment than the one studied in the first contribution. In the second environment, to gain access to some items, not only may other items need to be moved out of the way, but also the overall number of movements for items within the system will be highly affected by the number of items stacked over each other. In addition, the searcher is given a task that includes searching and retrieving a set of items, rather than just one item.

For the second contribution, the performance of the search heuristic is analyzed through a Statistical Design of Experiments, and it was found that searching and retrieving multiple items instead of just a single item, would decrease the variability in search times for each storeroom configuration.

Finally, in chapter 6, conclusions of this research and suggestions for future research directions are presented.

ACKNOWLEDGMENTS

I would like to express my immeasurable appreciation to my advisor and mentor Dr. Jennifer Pazour, for her support, valuable comments and suggestions that benefited in the completion of this research. Without her help this dissertation would not have been possible.

I extend my deepest gratitude to Dr. Ahmad Elshennawy and Dr. William Thompson, not only for being members of my committee, but for their support throughout my career as a graduate student at UCF and for putting their faith in me.

I am grateful to my committee member Dr. Steven Leon for his positive feedback and for all the conversations we had together which I greatly enjoyed and benefited from.

I am thankful to all the people who were kind enough to provide me with their support and friendship. Specially, I would like to thank my family and my friends: Ahmed Bekhit, Ahmed Korba and Ahmed Hemida.

Finally, I would like to thank the Department of Industrial Engineering and Management Systems at UCF for their support to me as a graduate student. Special thanks to the Office of Naval Research that partially supported this research through the Young Investigator grant number N000141310594.

TABLE OF CONTENTS

LIST OF FIGURES x

LIST OF TABLES xi

LIST OF ABBREVIATIONS/ ACRONYMS xii

CHAPTER ONE: INTRODUCTION 1

References 6

CHAPTER TWO: LITERATURE REVIEW 7

Introduction 7

Sea Based Logistics 7

Dense Storage 13

Search Theory 16

Research Gaps 20

References 23

CHAPTER THREE: PROBLEM STATEMENT 26

Contribution 1: Search Plan for A Single Item in an Inverted T *K*-Deep Storage System . 26

Contribution 2: Search Plan for Multiple Items in an Inverted T Dense Storage System ... 28

CHAPTER FOUR: SEARCH PLAN FOR A SINGLE ITEM IN AN INVERTED T K-DEEP STORAGE SYSTEM 31

Abstract 31

Introduction 31

Literature Review 36
Dense storage 36
Search theory 38
Problem Formulation 40
Travel Time 44
Repositioning Time 45
Repositioning Procedure 50
Put-back Time 51
Search Plan Heuristic 52
Operational Search Plan 53
Numerical Experiments 55
Conclusion and Future Research 65
References 67
CHAPTER FIVE: SEARCH PLAN FOR MULTIPLE ITEMS IN AN INVERTED T DENSE STORAGE SYSTEM 70
Abstract 70
Introduction 70
Literature Review 75
Problem Formulation 77
Travel Time 82
Repositioning Time 82

Put-back Time 84

Retrieval Time 86

Search Plan Heuristic 87

Expected Search and Retrieval Times 88

Search Plan 90

Numerical Experiments 92

Conclusion and Future Research 100

References 102

CHAPTER SIX: CONCLUSION AND FUTURE RESEARCH 103

Contributions 103

Contribution 1 104

Contribution 2 105

Conclusions 106

Future Research 107

APPENDIX A: CONTRIBUTION 1 EXPERIMENTAL RESULTS 109

APPENDIX B: CONTRIBUTION 2 EXPERIMENTAL RESULTS 145

LIST OF FIGURES

Figure 1: Sea Basing Logistics Architecture Overview (McCarthy, 2005) 8

Figure 2: Present MPF(F) Squadrons Locations (FAS, 2000) 8

Figure 3: Classical detection search problems in discrete space and time. The problems gain complexity with additional features and constraints, indicated by the red plus signs (Lau, 2007) 19

Figure 4: Components of the search procedure 33

Figure 5: Sample inverted T system with $\boldsymbol{k} = \mathbf{2}$ and $\boldsymbol{h} = 6$. 42

Figure 6: Repositioning outside the storeroom grid 49

Figure 7: Search Time versus the Number of Pallets for a storeroom with $\boldsymbol{k} = \mathbf{2}$ and $\boldsymbol{h} = \mathbf{10}$ for repositioning policy 1 57

Figure 8: Search Time versus the Number of Pallets for a storeroom with $\boldsymbol{k} = \mathbf{2}$ and $\boldsymbol{h} = \mathbf{10}$ for policy 2 58

Figure 9: Components of the search procedure 72

Figure 10: Side view of the k-deep s-stacked dense storage system 73

Figure 11: Sample Inverted T system with $\boldsymbol{k} = \mathbf{2}$ and $\boldsymbol{h} = \mathbf{6}$ 79

Figure 12: Main effects plots for expected search time 97

Figure 13: Interaction effect plots for expected search time 98

Figure 14: Main effects plots for standard deviation 99

Figure 15: Interaction effect plots for standard deviation 99

LIST OF TABLES

Table 1: Classes of Supply Onboard a Sea Base (Cook & Hicks, 2005).....10

Table 2: Literature Review Matrix 21

Table 3: Expected Search Time and Standard Deviation of Search Times in Time Units for repositioning policies 1 and 2 with different *k* and *h* values.....59

Table 4: A-D test for storeroom with $\boldsymbol{k} = \mathbf{2}$ and $\boldsymbol{h} = \mathbf{6}$ 62

Table 5: Mean $\boldsymbol{\mu}$ and standard deviation $\boldsymbol{\sigma}$ for systems with search time probability distribution fitting a lognormal distribution 63

Table 6: Values for each factor level 93

Table 7: Response for each of 10 replications 94

Table 8: Main and interaction effects and corresponding p-vales for expected search time ...95

Table 9: Main and interaction effects and corresponding p-vales for standard deviation 95

LIST OF ABBREVIATIONS/ ACRONYMS

AF	Amphibious Force
ANOVA	Analysis of Variance
AS/RS	Automated Storage and Retrieval Systems
C	Set of cells in the storeroom
C_O	Set of occupied cells
C_U	Set of unoccupied cells
C_O^{BC}	Set of cells in the bottom center part of the storeroom
C_O^{BL}	Set of cells in the bottom left part of the storeroom
C_O^{BR}	Set of cells in the bottom right part of the storeroom
C_O^{UL}	Set of occupied cells in the upper left part of the storeroom
C_O^{UR}	Set of occupied cells in the upper right part of the storeroom
cdf	Cumulative distribution function
CJOS COE	Combined Joint Operations from the Sea Center of Excellence
CLF	Combat Logistics Force
CONUS	Continental United States
DoD	Department of Defense
DOE	Design of Experiments
E	Expected search time
E_M	Expected search time for M pallets
E_ρ	Expected retrieval time for a single pallet
E_{ρ_M}	Expected retrieval time for M pallets
GFSMP	Global Fleet Station Mission Planner

h	Storeroom length
H/W	Shape factor
I/O	Input/ Output
i	Index of cell i
j	Index of cell j
k	Accessibility constant
LCAC	Landing Craft Air Cushion
M	Number of pallets to be retrieved from the storeroom or the number of pallets in an order
MEB	Marine Expeditionary Brigade
MPF(F)	Maritime Prepositioning Force (Future)
N	Total number of potential storage locations in the horizontal direction
NAVSEA	Naval Sea Systems Command
n	Number of cells
N'	Total number of potential storage locations in both horizontal and vertical directions
n'	Total number of pallets in a stacked dense storage system
ONR	Office of Naval Research
OSP	Optimal Search Path
P_b	Put-back time
p_i	Probability mass function
R	Repositioning time
$Repo$	Array of already repositioned pallets

RO/RO	Roll-on, roll-off
SD	Standard deviation of search times
SD_M	Standard deviation of search times for M pallets
SD_{ρ_M}	Standard deviation of retrieval times of M pallets
SKU	Stock Keeping Unit
s	Stack height
T_i	Search time
u	Index of cell u representing the closest open aisle location to cell i
VHD	Very high density
v	Index of cell v representing the closest open aisle location to cell j
W	Storeroom width
α	Counter of number of put-back procedures
μ	Mean
ρ	Retrieval time
σ	Standard deviation
τ	Travel time
ς	Actual number of items stacked over each other
Ω	Unordered array of requested pallets

CHAPTER ONE: INTRODUCTION

Dense storage provides a means to use storage space efficiently in warehousing and distribution of goods. In a dense storage environment, the storage density is increased through stacking a lane with several pallets deep on the floor and several pallets high (Gue, 2006). However, a trade-off between storage density and labor efficiency arises when a dense storage system is adopted. In a dense storage system, not all of the stored items or pallets are directly accessible to the order picker, therefore, retrieval and storage requests require more effort as some of the pallets need to be moved out of the picker's way to gain access to other deeply stored items.

The work of Gu, Goetschalckx, and McGinnis (2007) provides a good survey of warehousing literature. To our knowledge, only a few contributions have been made in the regards to stored items' locations uncertainty. Partial or complete uncertainty about stored items' locations in a storeroom can arise from different reasons depending on the application, such as the initial absence of a record of the stored items' locations when the items were stowed in the storeroom, or the absence or the wear of identification tags on the pallets or cases, or even the propagation of uncertainty over time, where items' locations are known initially, but due to the continuous movement, repositioning and retrieval of items, one starts to lose track of the items' locations. When items' locations are unknown, a search effort has to be made in order to find the requested item or group of items before retrieval. In order to control this search effort the order picker or the searcher has to follow a search plan usually with the objective of decreasing the time required to find a certain item or group of items.

So far scientific research has focused on studying dense storage systems with automated storage and retrieval systems (AS/RS). However, 80-83% of the existing warehouses still operate using manual order picking systems, where a picker walks or drives to the goods

(picker-to-part) and manually retrieves the items and transports the items from the storage location to the output point (De Koster, Le-Duc, & Roodbergen, 2007).

Therefore, we are studying a dense storage environment operated manually with uncertainty about stored items' locations and developing search policies to retrieve either a single item or multiple items from such a storeroom. Such storage environments can be encountered in many applications, such as space, urban and sea-based logistics applications.

Over the years, research efforts addressing "space logistics" have increased with the increase in attention given to space exploration (Lee et al., 2008). Stowage of commodities in attempt to sustain future space missions or even space outposts, has to be made in a way that uses the scarce storage spaces onboard of the envisioned space ships efficiently (Siddiqi & De Weck, 2007). In addition, retrieval of such stored commodities has to be made in a selective manner and in minimal time. One way to efficiently use the available space is through adopting dense storage systems. However, the complexity of the dense storage environment onboard such space craft might increase if there is either partial or complete uncertainty about stored items' locations. Such an uncertainty about items' locations is predicted to increase during the long periods of time through which those missions will operate due to the continuous movement and retrieval of stored items and thus the accompanying change of items' storage locations. Therefore, problems arising from the use of dense storage in the space logistics application, can be modeled as a manual dense storage environment with possible uncertainty about stored items' locations and a search plan has to be developed in order to assist the picker or the astronaut in finding and retrieving a certain item onboard the space ship. Yet, a few issues have to addressed and studied by experts and engineers in the space logistics field, such as the overall dimensions of storerooms and the dimensions of stored items or pallets onboard the space ships,

the manning level and the number of people who will be dedicated to the sustainment processes including storing, searching and retrieval of items, and finally, the possible configuration of such dense storage system.

Another application in which dense storage is advantageous, is "urban or city logistics". Urban logistics is the field that studies goods' movements within an urban environment and how they are affected by urban traffic and topography particularities (Muñuzuri, Larrañeta, Onieva, & Cortés, 2005). Urban logistics is generally concerned with sustaining today's cities and their residents, such sustainment requires freight transportation. However, urban freight transportation usually has its negative impacts, such as increased traffic levels, elevating levels of pollution and declining levels of profitability due to increasing fuel prices in the recent years (Anderson, Allen, & Browne, 2005). One way to overcome the problems accompanied with urban freight transportation is through having distribution centers located closer to city centers and capable of sustaining the city for an extended period of time. The construction of such distribution centers will cost more than traditional centers located outside the cities, especially in terms of the cost of space. Therefore, such expensive storage space should be used efficiently, and one way to do that is through adopting a dense storage system. Uncertainty about stored items' locations might arise in such dense systems in such application for different reasons as discussed in the space logistics application. Therefore, the order picker in such a dense and an uncertain storage system should be equipped by a search plan for an item or for multiple items in order to find and retrieve the requested item or items.

Several storage environments can be modeled as a dense storage environment with stored items' locations uncertainty as we discussed in the space and urban logistics applications; however, we are mainly motivated in this work by one of the US Navy's fundamental concepts

for the 21st century, which is the Sea Basing concept. The main purpose behind having a sea base is to minimize the logistics footprint ashore while maintaining the capability of providing logistical sustainment of forces ashore participating in a variety of conflicts or missions (Clark, 2002). A sea base consists of several ships, some of which have storage spaces, and the whole cluster of ships acts as a maritime distribution center. This floating distribution center will utilize the concept of dense storage, i.e. ships' storerooms are configured in a way that allows the maximization of available cargo onboard. However, the ships of the sea base must have selective offloading capability, which is the ability to locate and deliver any piece of equipment or commodity on board to support the forces ashore (Futcher, 2003). The problem of stored items' locations uncertainty arises as well in such a dense storage environment. Consequently, a search effort has to be exerted and a search plan has to be followed in order to find an item or multiple items before retrieval to a specific output location.

In this research we are studying dense storage environments operated manually. Adding to the complexity of the studied problem, we assume that the exact locations of stored items are completely unknown to the order picker. Generally, in a dense storage environments, stored items have to be moved around or repositioned to other locations in order to gain access to other deeply stored items. Not knowing exact items' locations requires searching for the requested item while moving things out of the way which increases the problem's complexity. We are developing search plans for either a single item or multiple items in a dense storage environment. For each storeroom configuration the search plan when followed results in a search sequence, and for each search sequence, search times for items are calculated through derived search time equations. The main objective behind developing a search plan is lowering the time needed on average to find an item or a group of items in a certain storeroom with a

specific configuration, or the expected search time. In addition to the expected search time, the standard deviation of search times is also calculated for each storeroom configuration.

Several statistical techniques are used as well to analyze the experimental results. Such as deriving the probability distribution of search times for storerooms with small configurations, Design of Experiments (DOE) and Analysis of Variance (ANOVA).

The rest of this document is organized as follows: in Chapter 2: we review related relevant literature on the sea basing concept, dense storage and search theory. In Chapter 3, details of the studied problems and their problem statements are presented. In Chapter 4, the first research contribution is presented in detail. The problem of searching for a single item in a dense storage environment with inverted T configuration is modeled, search plan is proposed and search times for items are calculated. The resulting search times are further analyzed and conclusions and insights are provided. The second research contribution is presented in Chapter 5, in which the problem of searching for multiple items not just a single item in an inverted T dense storeroom. In this second contribution, we assume having s pallets stacked over each other instead of just having one row of pallets stacked on the floor. The performance of the proposed search plan heuristic in this chapter is tested through design set of experiments and the results are analyzed using the Analysis of Variance (ANOVA). In Chapter 6, main contributions of this research, conclusions, and recommendations for future work are presented. Finally, Appendix A and Appendix B exhibit the numerical results of contribution 1 and contribution 2 respectively.

References

Anderson, S., Allen, J., & Browne, M. (2005). Urban logistics—how can it meet policy makers' sustainability objectives? *Journal of Transport Geography, 13*(1), 71-81.

Clark, A. V. (2002). *Sea Power 21 Series, Part I: Projecting Decisive Joint Capabilities*.

De Koster, R., Le-Duc, T., & Roodbergen, K. J. (2007). Design and control of warehouse order picking: A literature review. *European Journal of Operational Research, 182*(2), 481-501.

Futcher, F. W. (2003). Selective Offload Capability Simulation (SOCS): An Analysis of High Density Storage Configurations: DTIC Document.

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

Gue, K. R. (2006). Very high density storage systems. *IIE Transactions, 38*(1), 79-90.

Lee, G., de Weck, O., Armar, N., Jordan, E., Shishko, R., Siddiqi, A., & Whiting, J. (2008). *SpaceNet: Modeling and Simulating Space Logistics.* Paper presented at the AIAA Space 2009 Conference and Exposition.

Muñuzuri, J., Larrañeta, J., Onieva, L., & Cortés, P. (2005). Solutions applicable by local administrations for urban logistics improvement. *Cities, 22*(1), 15-28.

Siddiqi, A., & De Weck, O. L. (2007). Spare parts requirements for space missions with reconfigurability and commonality. *Journal of Spacecraft and Rockets, 44*(1), 147-155.

CHAPTER TWO: LITERATURE REVIEW

Introduction

In this chapter, we survey the literature related to the problem under investigation here. We start by surveying the literature of U.S. Navy's Sea Basing concept, which is considered as the main motivation of this work. Then a survey of literature related to dense storage. A review of the search theory related to our problems is provided. Finally, a summary of research gaps is presented.

Sea Based Logistics

In 2002 The U.S. Navy released "Sea Power 21" highlighting its vision of how to meet the new century's challenges. One of the core concepts of the Navy's vision is Sea Basing. The main purpose behind having a sea base is to minimize the logistics footprint ashore while maintaining the capability of providing logistical sustainment of forces ashore participating in a variety of conflicts or missions (Clark, 2002).

Figure 1 provides an overview of the Sea Basing Logistics architecture. The architecture is comprised of three main nodes: continental United States (CONUS), advanced base, and the sea base. CONUS and the advanced base represent the primary logistics nodes. The majority of all sustainment for initial resupply for ground units originates in CONUS. The advanced base is central to the support of the sea base because it is the primary warehouse and transshipment point through which the majority of all supplies needed by the sea base will pass. The sea base is the primary demand node since it supports and sustains expeditionary ground forces projected ashore. A network of logistics shuttle ships and aircraft connect the sea base with the advanced base (McCarthy, 2005).

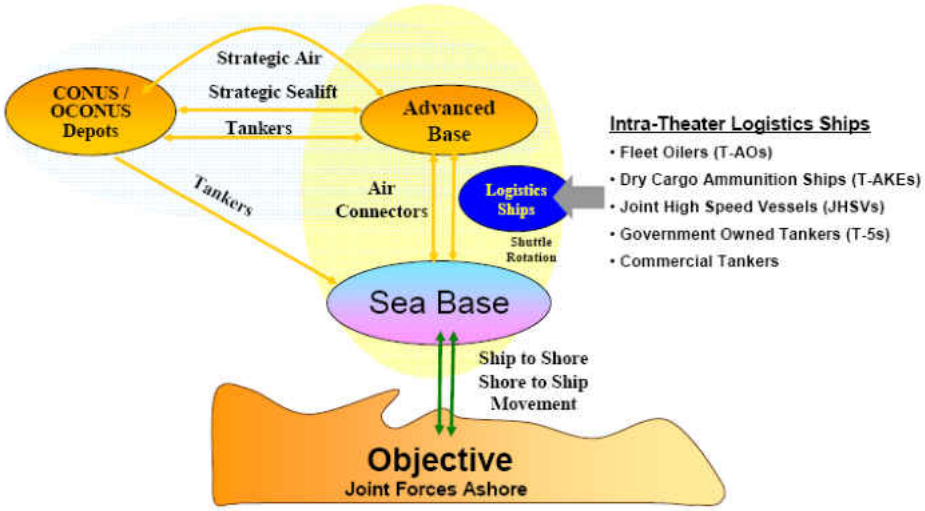

Figure 1: Sea Basing Logistics Architecture Overview (McCarthy, 2005)

A sea base can consist of numerous platforms such as Carrier Strike Groups, Expeditionary Strike Groups, an Amphibious Force (AF), and other Combined/Joint naval platforms. However, the most integral part of the sea base is the Maritime Prepositioned Force (Future) (MPF(F)) ships. MPF(F) squadrons are prepositioned in key locations around the globe as shown in Figure 2. These squadrons will enable a 30-day logistical sustainment to a Marine Expeditionary Brigade (MEB) participating in a variety of conflicts (FAS, 2000).

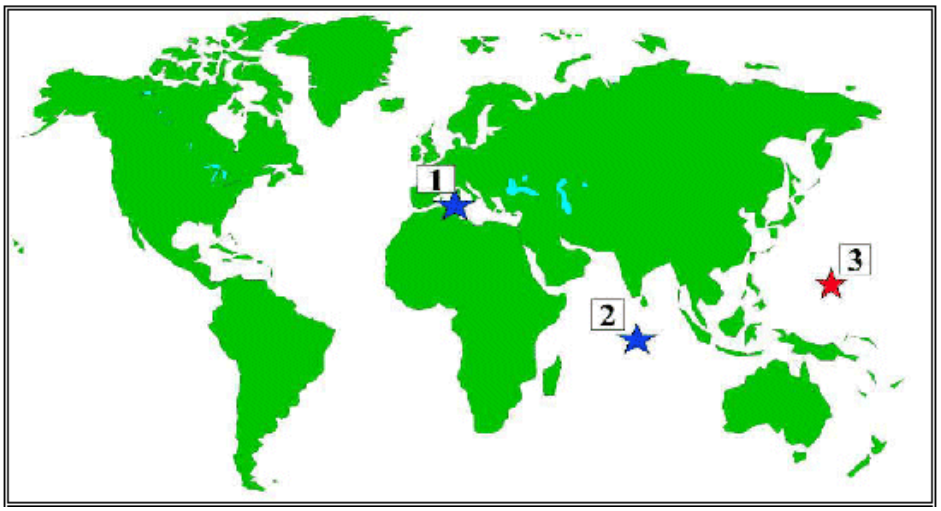

Figure 2: Present MPF(F) Squadrons Locations (FAS, 2000)

Several research efforts have been made to study the Sea Basing Logistics architecture, such as Button, Gordon, Hoffmann, Riposo, and Wilson (2010) who provide simulation analyses in order to suggest possible compositions of the MPF(F) squadrons that can provide the necessary logistical sustainment. And Button, Gordon IV, Riposo, Blickstein, and Wilson (2007) who examine how the Sea Basing Concepts could be applied to joint operations of the Navy, Army and the Marine Corps.

Gue (2003) proposes an optimization model for the location of sea-based supply units for small, highly mobile US Marine Corps combat teams. Gue's model can be used in determining the feasibility of a given logistical plan, in addition, it can be used in the development of operational logistical plans in real time.

Brown and Carlyle (2008) develop an integer linear program with the objectives of minimizing shortages and maximizing utilization of transports and total volume delivered by a specified combat logistics force (CLF). The model they developed can be used by fleet planners in the process of conducting comparative analyses when used in combination with other organizational tools.

Since a sea base can be used in humanitarian missions, Salmerón, Kline, and Densham (2011) develop a planning tool named Global Fleet Station Mission Planner (GFSMP) that can used to explore the feasibility of a US Naval humanitarian engagement plan. GFSMP tool is developed to substitute manual planning of humanitarian missions. GFSMP is used to optimize the route and schedule in order to maximize the total mission value earned.

In order for it to support a MEB ashore, a sea base has to carry ten different, well-defined, classes of supply shown in Table 1 (Cook & Hicks, 2005).

Table 1: Classes of Supply Onboard a Sea Base (Cook & Hicks, 2005)

CLASS	SUPPLIES
I	Subsistence and comfort items
II	Clothing, individual equipment, and administrative supplies and equipment
III	Petroleum fuels and lubricants
IV	Construction and barrier materials
V	Ammunition
VI	Personal demand items
VII	Major end items
VIII	Medical material
IX	Repair parts
X	Material to support nonmilitary programs
Misc.	Water Salvage, and captured material

These materials can be provided to the MEB in bulk or individual marine units can require custom-tailored packages depending on the forecasted needs. Therefore, MPF(F) must have a selective offloading capability, where different pallets from different classes of materials shown in Table 1, are picked up quickly and efficiently from the ships' storerooms and transferred to the flight decks to be picked up by carriers to be delivered to forces ashore. Selective offloading is the ability to offload any type of equipment or cargo any time with any quantity (Futcher, 2003).

Without selective offloading, the current dependence on shore staging of materials will remain the same, accordingly the need of manpower to break supplies ashore and supply forces will remain the same.

In terms of selective offloading, cargo can be further divided into two categories. The first one is selective offloading of roll-on, roll-off (RO/RO) cargo, and the second category is selective offloading of dry cargo.

From its name, roll-on, roll-off (RO/RO) cargo is the type of cargo that can roll out of the sea base ships' storerooms, such as vehicles, tanks, and trucks. RO/RO cargo is generally

selectively offloaded in a tactical way which allows the arrival of both the personnel and their equipment of the RO/RO type at the same time. RO/RO equipment are generally delivered to forces ashore after they have been accessed in the ships' storerooms, repositioned, and delivered to the shipboard staging area before being delivered via Landing Craft Air Cushion (LCACs), vertical lift aircrafts, or via RO/RO ramps.

Up to our knowledge, research is being currently conducted to investigate the selective offloading processes of RO/RO cargo, in which the storage and retrieval processes will be modeled as a rush hour puzzle problem. However, the rush hour puzzle problem is PSPACE-complete problem and very few efforts have been made in order to solve such a complex problem, such as the use of Genetic Programming heuristics to solve a few cases of that problem as in Hauptman, Elyasaf, Sipper, and Karmon (2009).

However, in this research the term selective offloading will refer to the process of accessing, retrieving, and delivering of items that belong to the second type of cargo, i.e. dry cargo. These items are densely packed in ships' storerooms to maximize storage utilization and they will be delivered in palletized form to sustain forces ashore (Button et al., 2010).

The problem of selective offloading is always coined with the problem of increased manning levels. Generally, selective offloading of a specific item from the dry cargo type stored onboard a sea base ship is labor intensive, as significant amount of labor will use material handling equipment such as forklifts or even move things by hand to gain access to a specific item and to retrieve it before it is delivered. Another thing that adds to the problem's complexity, is the dense packing or storage of cargo onboard of the ships in order to efficiently utilize the available space.

The Navy finds the reduction of manning level crucial to the sea base operation. Therefore, research funded by the Naval Sea Systems Command (NAVSEA) and the Office of Naval Research (ONR) has focused on developing systems and technologies with three features: 1) provide selective offloading capability, 2) utilize the available storage space efficiently through the use of dense storage, and 3) reduce manning level (Futcher, 2003).

Therefore, previous research focused on the use of Automated Storage and Retrieval Systems (AS/RS) onboard of sea base ships in order to move, position, reposition and retrieve materials from the storerooms, while 1) doing all of these processes in the lowest possible time (selective offloading), 2) having dense storerooms, and 3) using the lowest possible manning level.

For example, Futcher (2003) studies the trade-off between storage density and mean retrieval time in dense storage environments with different densities utilizing AS/RS to be implemented onboard sea base ships' storerooms. Gue and Kim (2007) develop models for "physical goods storage system based on 15-puzzle". Gue and Kim (2007) analyze the NAVSTORS system using their models. NAVSTORS[1] system utilizes "moving floor" mechanism to move stacks of pallets and containers stowed in ships' storerooms to a specific input/ output location in order to be transferred to the point from which these pallets or containers will be delivered to forces ashore.

Similar to Futcher (2003), Gue and Kim (2007) assume the use of AS/RS onboard of sea base ships, and focus on studying the effect of increasing the storage density on retrieval times of stored items. Both works assume the use of dense storage in order to efficiently utilize the available and expensive space onboard of sea base ships. However, they chose to increase the

[1] http://www.agilesystems.com/navstors_frame.htm

storage density through two different types of dense storage configurations. Futcher (2003) uses an inverted T configuration, while Gue and Kim (2007) use a slide-puzzle architecture. Details about these two different configurations will be provided in following sections.

In this work we build on what have been done before. We model a dense storage environment with an inverted T configuration as the one studied by Futcher (2003), with possible future use onboard of sea base ships, therefore, must have selective offloading capability of items. However, unlike the work of Futcher (2003) and Gue and Kim (2007), we do not assume the usage of AS/RS, on the contrary, we assume the use of manual storage and retrieval processes not automated ones. We add to the complexity of the problem under study, by assuming not knowing exact items' locations in the storeroom, therefore, one must "search" for requested item or group of items before being retrieved.

Therefore, before introducing the studied problems, two other sets of literature have to be surveyed: 1) dense storage literature, and 2) search theory literature.

Dense Storage

Gu, Goetschalckx, and McGinnis (2007) provide an extensive review on warehouse operation planning problems. In their work, research papers are classified according to the warehouse functions, i.e., receiving, storage, order picking, and shipping.

Research efforts summarized in Gu et al. (2007) have one main thing in common which is the assumption that all stored items' locations are known with certainty. In addition, most of the papers surveyed assumed the use of automated storage and retrieval systems. All of the papers surveyed in that work do not assume any degree of uncertainty about storage items' locations, which is one of the main assumptions of our work.

The rapid development of business in the recent years caused an increase in the value of facility space, the thing that forced many firms to think about utilizing the available storage space efficiently. One way to improve storage density is through the use of dense storage systems (Gue, 2006). In a dense storage system, pallets are stacked on the floor in a lane that could be several pallets deep, and maybe two or more pallets high.

Some of the publications, such as Rana (1990), developed ways to increase the storage density in warehouses through dedicating less aisle space by having narrow or very narrow aisles. Implementation of such systems usually result in problems such as travel congestion or even restricted travel in one way only of the aisle. Parikh and Meller (2010) discussed a few of the problems associated with the use of these systems.

Another way to increase storage density is through having *k*-deep pallet racks, where each location contains *k* pallets behind each other, or through creating block stacking areas, in which pallets for a single stock keeping unit (SKU) are stacked on the floor in a lane that could be several or *k* pallets deep and two or more pallets high. Such systems are always designed such that each lane has the same SKU, accordingly, no moving is required of one item to access the other. An example of the *k*-deep systems, is the double-deep automated storage and retrieval systems discussed by Lerher, Sraml, Potrc, and Tollazzi (2010), in addition, Matson, Sonnentag, White, and Imhoff (2014) provide an extensive review of the problems associated with block stacking.

The work presented here falls under the area of very high density (VHD) storage system. According to Gue (2006), a storage system is said to have very high density (VHD) when "it is sometimes necessary to move interfering items in the system in order to gain access to desired items". In a *k*-deep very high density storage system, one might have to move up to *k-1* pallets

in order to gain access to the desired pallet. For instance, in a 2-deep or double-deep system, at most one pallet has to be moved in order to retrieve the one behind it. However, a single-deep storage system is not considered a VHD system, as all pallets are accessible directly from aisles.

In this research the constant k will be referred to as the accessibility constant, which is according to Futcher (2003) "a variable directly related to the accessibility or physical access of any particular pallet" in a dense storage system.

The main difference between existing VHD storage systems' literature and this work is that existing literature always assumed the existence of perfect information about stored items' locations. However, some of the existing work was motivated by the Sea Basing concept and the need of dense storage onboard of sea base ships, such as the work of Futcher (2003) and Gue and Kim (2007).

Futcher (2003) studies dense storage environments with selective offloading capability onboard of sea base ships with an inverted T configuration. An inverted T storage configuration has one horizontal and one vertical aisle, and provide the lowest mean retrieval times with a certain fixed value of the accessibility constant k as has been proved by Gue (2006). The main purpose of Futcher's (2003) research is the study of the trade-off between storage density and the mean retrieval time for different inverted T storage configurations. Locations of stored items in this research are assumed to be known and no search effort is needed to allocate a certain item in the storeroom before it is retrieved, which is considered the main difference between Futcher's work and our work.

Gue (2006) studies deep bulk-storage systems. In a deep bulk-storage system one might require to move several items in order to be able to retrieve a single item. Moving items around in order to gain access to a certain item entails the change of items' locations over the time, but

that does not necessarily mean that the exact location of the requested item is not known before it is being retrieved, therefore, search for the requested item does not occur as it does in our work. Gue's (2006) results show that in order to achieve the highest storage density, a storage system should be arranged with several aisles connected by a single cross-aisle.

Another work motivated by the Sea Basing concept is the work of Gue and Kim (2007). Gue and Kim (2007) firstly propose an optimal algorithm for a puzzle-based storage system with a single open cell. Secondly, they investigate the relationship between storage density and expected retrieval time for puzzle-based and aisle-based systems. Similar to the work of Futcher (2003) and Gue (2006), the work of Gue and Kim (2007) did not investigate the possibility of having no information about exact stored items' locations, consequently, the possible need to spend some amount of time to search for the requested item before it is retrieved.

Existing dense storage literature might share the same motivational background as our work, and might provide a good basis for this research effort specially in terms of storeroom configurations; however, there are still some research gaps that need to be filled and studied, such as the use of manual storage and retrieval systems and the assumption of not knowing exact stored items' locations. Further discussion about gaps in existing research gaps will be presented in the research gaps section, following a survey of existing literature of the search theory related to our research.

Search Theory

Search theory is concerned with the problem of how to allocate a limited amount of resources to find a target (or targets) whose location is not precisely known, such that the reward as specified by the measure of effectiveness is maximized (Frost & Stone, 2001).

Search theory originated from the work of Koopman (1956) and others during World War II. In addition to its application in naval operations, search theory has been applied to other problems such as rescue operations of the U.S. Coast Guard, land search conducted by U.S. Air Force, fishing problems, police patrol analysis, and oil exploration (Lau, 2007; Stone, 1983).

Up to our knowledge, the only work where search theory has been applied in warehousing problems is the work of Hariharan and Bukkapatnam (2009). They study the impact of using a poorly designed RFID system to locate a "misplaced item" in a warehouse, and how such a looking a misplaced item in a warehouse with a poorly designed RFID system will "actually cost the firm more than without it".

Lau (2007) categorized the searcher's or responder's knowledge about the target location regardless of the application into three categories:

- Perfect knowledge: the responder knows where the target is at all times,
- Imprecise knowledge: the responder has some idea of where it could be, but cannot be sure,
- No knowledge: the target could be anywhere.

In case of having perfect knowledge about the target's location, the responder will simply locate or intercept the target through the most direct path. On the other hand, if the target's location is unknown, the best that can be done is to cover the entire area of interest as quickly as possible.

The elements of the basic optimal search problem are (Koopman, 1980; Lau, 2007; Stone, 1983):

- Target information: represented by a prior probability distribution of its location and other relevant states,
- Searcher capability: the amount of search effort available for use and the restrictions on the manner in which it can be deployed,
- Detection function: a function relating the amount of search effort applied at a location to the probability of detecting a target that is actually there,
- Measure of search effectiveness: an objective that defines a reward in relation to the probability of finding the target when a given search plan is executed.

Benkoski, Monticino, and Weisinger (1991) in their literature review of the search theory problems, classified the types of problems into two broad categories:

- One-sided search problems, and
- Two-sided search games.

In one-sided search problems it is assumed that "the target is unwilling or unable to respond to the searcher's action, such that once the search process has started, it is only the searcher's plan that affects the anticipated outcome". As for the two-sided search games, it is assumed that "the searcher and target cooperate to find each other, or where the target actively seeks to hide from the searcher" (Lau, 2007).

Figure 3, from Lau (2007), outlines the main one-sided detection search problems addressed in search theory literature.

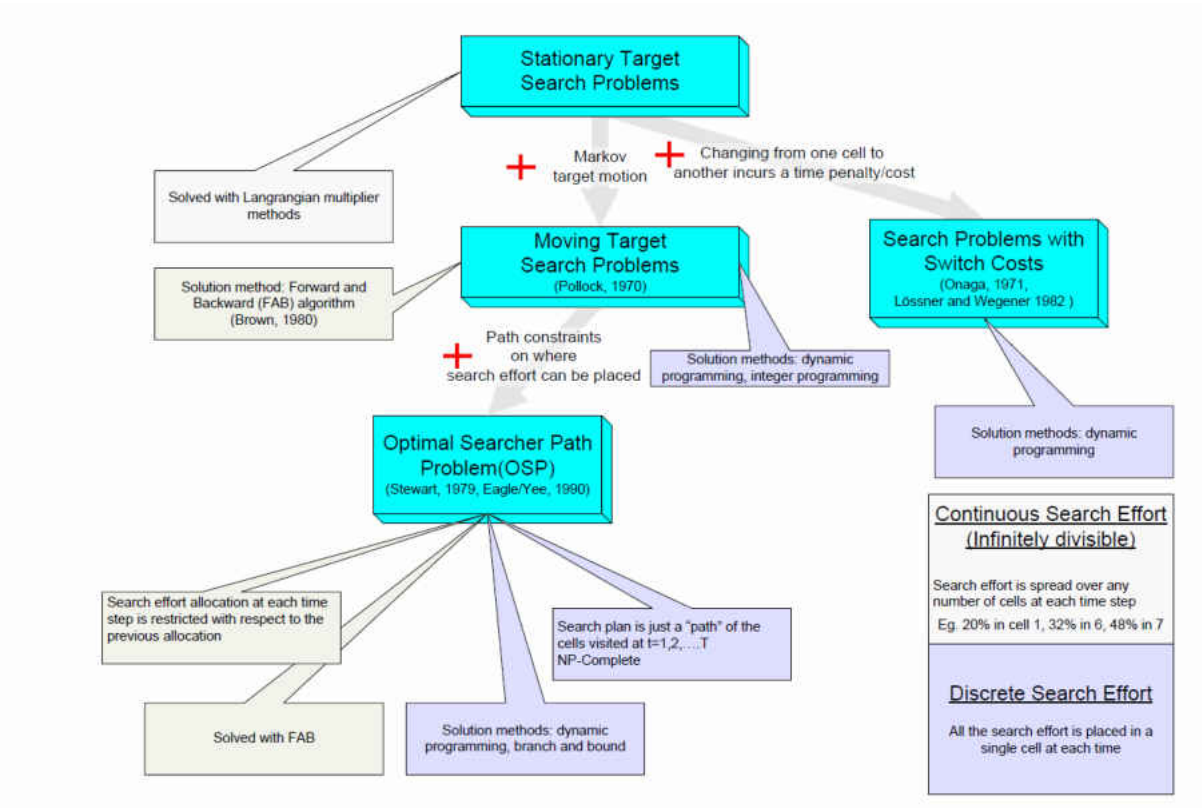

Figure 3: Classical detection search problems in discrete space and time. The problems gain complexity with additional features and constraints, indicated by the red plus signs (Lau, 2007)

For the purpose of this work, the problem of searching for an item with imprecise knowledge about its location in a dense storeroom onboard a sea base vessel can be classified as a one-sided detection search problem with stationary target (which is the item), a single searcher (the sailor), and with the objective of developing a search plan that lowers the expected time for detection, given the knowledge of:

- The environment structure: the storeroom configuration
- Searcher capability: restrictions on movement path
- A priori target information: initial locations or probability distribution of its location

The stationary target search problem is the problem of finding a stationary target located at a point on a plane or alternatively in one of a number of discrete cells. The objective here is to

develop an optimal effort allocation among the cells of the search environment, in terms of the amount of effort to be placed in each cell such that the overall probability of detecting the target is maximized, or to minimize the expected detection time (Lau, 2007).

Two types of stationary target search problems can be found in literature. The first one is continuous effort stationary target search problems and the second one is discrete effort stationary target search problems. For the purpose of this work we will focus of the second type; however, for information on the first, the reader is referred to Stone (1975).

Discrete effort stationary target search problems consider the cases where all the available search effort is restricted to be in a single cell at each time. In these types of problems, search resources cannot be simultaneously deployed in multiple places; the optimization task then became that of finding the best sequence of cells for the searcher to visit. The reader is referred to Lau, Huang, and Dissanayake (2005) and Lau (2007) for more information about this problem and different solution methodologies.

Stone (1983) studied different elements of search planning. He presented an approach that encompasses both objective and subjective methods and provided compromises between theoretically optimal plans and operationally feasible ones. In this work we are developing an operational search plan, and the reason that we are interested in developing an operational plan can be attributed to the fact that we are studying a manual environment and it is not always possible to provide human searchers with optimal search paths to follow.

Research Gaps

This chapter helps in identifying several research gaps and possible research directions. Table 2 provides a summary of the surveyed literature and where this research fits in.

Table 2: Literature Review Matrix

	Sea Based Logistics		**Dense Storage**		**Search Theory**	
Author (Year)	Concept/ Architecture	Mathematical Models	Inverted T	Other	Obstacles	No Obstacles
Koopman (1956, 1980)						✓
Stone (1975, 1983)					✓	✓
Rana (1990)				✓		
Benkoski et al. (1991)					✓	✓
FAS (2000)	✓					
Frost and Stone (2001)						✓
Clark (2002)	✓					
Gue (2003)		✓				
Futcher (2003)		✓	✓			
Cook and Hicks (2005)	✓					
Lau et al. (2005)						✓
McCarthy (2005)	✓					
Gue (2006)			✓	✓		
Button et al. (2007)	✓					
Gu et al. (2007)				✓		
Gue and Kim (2007)				✓		

Lau (2007)						✓
Brown and Carlyle (2008)		✓				
Hariharan and Bukkapatnam (2009)					✓	
Hauptman et al. (2009)		✓		✓		
Button et al. (2010)	✓					
Lerher et al. (2010)				✓		
Parikh and Meller (2010)				✓		
Salmerón et al. (2011)		✓				
Matson et al. (2014)				✓		
This Research		✓	✓		✓	

In terms of research gaps, first of all, in terms of the Sea Basing application, it seems that the use of dense storage is crucial to the whole Sea Basing operational concept due to the high value of the limited storage space onboard of the sea base ships. Research efforts such as Futcher (2003) and Gue and Kim (2007) tried to provide solutions to apply dense storage in the Sea Basing environments. However, both efforts assumed the use of AS/RS and that exact stored items' locations are known before an item is retrieved. In this work we are relaxing these two assumptions, by assuming the use of manual storage and retrieval system and having limited to no information about exact stored items' locations, while having a dense storage environment.

Secondly, in terms of the dense storage literature, up to our knowledge, no previous research effort addressed the issue of not knowing the location of the item to be retrieved, consequently, no one addressed the subject of searching for an item in a dense storeroom before retrieving it,

especially, when this searching might involve moving things around in the storeroom in order to gain access to deeply stored items. Moving things around and repositioning them to gain access to a specific item is a complicated and time consuming process and adds, to the retrieval time of a certain item, the time required to reposition items out of the searcher's way and the time required to put-back items in order not to block the searcher's way and to maintain the storeroom configuration.

Finally, as for the search theory literature, and up to our knowledge, no effort has been made to develop either an optimal or operational search plan in a warehousing application, especially if this search effort will involve the searcher moving things out of the way in order to gain access to other items.

References

Benkoski, S. J., Monticino, M. G., & Weisinger, J. R. (1991). A survey of the search theory literature. *Naval Research Logistics (NRL), 38*(4), 469-494.

Brown, G. G., & Carlyle, W. M. (2008). Optimizing the US Navy's combat logistics force. *Naval Research Logistics (NRL), 55*(8), 800-810.

Button, R. W., Gordon, I., Hoffmann, D., Riposo, J., & Wilson, P. A. (2010). *Maritime Prepositioning Force (Future) Capability Assessment: Planned and Alternative Structures*. Retrieved from http://goo.gl/il3r5y

Button, R. W., Gordon IV, J., Riposo, J., Blickstein, I., & Wilson, P. A. (2007). Warfighting and Logistic Support of Joint Forces from the Joint Sea Base.

Clark, A. V. (2002). *Sea Power 21 Series, Part I: Projecting Decisive Joint Capabilities*.

Cook, H. B., & Hicks, D. M. (2005). *A Functional Architecture for a Logistics Expert System in a Sea Based Environment*. Retrieved from http://goo.gl/ImqOFs

FAS. (2000, Thursday, October 19, 2000). Sealift. Retrieved from http://fas.org/man/dod-101/sys/ship/sealift.htm

Frost, J., & Stone, L. D. (2001). *Review of search theory: advances and applications to search and rescue decision support*. Retrieved from http://goo.gl/oQPuHb

Futcher, F. W. (2003). *Selective Offload Capability Simulation (SOCS): An Analysis of High Density Storage Configurations*. Retrieved from http://goo.gl/uc8Z0U

Gu, J., Goetschalckx, M., & McGinnis, L. F. (2007). Research on warehouse operation: A comprehensive review. *European journal of operational research, 177*(1), 1-21.

Gue, K. R. (2003). A dynamic distribution model for combat logistics. *Computers & Operations Research, 30*(3), 367-381.

Gue, K. R. (2006). Very high density storage systems. *IIE Transactions, 38*(1), 79-90.

Gue, K. R., & Kim, B. S. (2007). Puzzle-based storage systems. *Naval Research Logistics (NRL), 54*(5), 556-567.

Hariharan, S., & Bukkapatnam, S. T. (2009). *Misplaced item search in a warehouse using an RFID-based Partially Observable Markov Decision Process (POMDP) model.* Paper presented at the Automation Science and Engineering, 2009. CASE 2009. IEEE International Conference on.

Hauptman, A., Elyasaf, A., Sipper, M., & Karmon, A. (2009). *GP-rush: using genetic programming to evolve solvers for the rush hour puzzle.* Paper presented at the Proceedings of the 11th Annual conference on Genetic and evolutionary computation.

Koopman, B. O. (1956). The theory of search. I. Kinematic bases. *Operations research, 4*(3), 324-346.

Koopman, B. O. (1980). *Search and screening: general principles with historical applications*: Pergamon Press New York.

Lau, H. (2007). *Optimal search in structured environments*: University of Technology, Sydney.

Lau, H., Huang, S., & Dissanayake, G. (2005). *Optimal search for multiple targets in a built environment.* Paper presented at the Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on.

Lerher, T., Sraml, M., Potrc, I., & Tollazzi, T. (2010). Travel time models for double-deep automated storage and retrieval systems. *International Journal of Production Research, 48*(11), 3151-3172.

Matson, J. O., Sonnentag, J. J., White, J. A., & Imhoff, R. C. (2014). *An Analysis of Block Stacking with Lot Splitting*. Retrieved from http://goo.gl/jLv8vI

McCarthy, J. D. (2005). Seabasing Logistics. Retrieved from http://www.dtic.mil/ndia/2005expwarfare/mccarthy.pdf

Parikh, P. J., & Meller, R. D. (2010). A note on worker blocking in narrow-aisle order picking systems when pick time is non-deterministic. *IIE Transactions, 42*(6), 392-404.

Rana, K. (1990). Order Picking in Narrow-Aisle Warehouses. *International Journal of Physical Distribution & Logistics Management, 20*(2), 9-15. doi:doi:10.1108/09600039010005133

Salmerón, J., Kline, J., & Densham, G. S. (2011). Optimizing schedules for maritime humanitarian cooperative engagements from a United States Navy sea base. *Interfaces, 41*(3), 238-253.

Stone, L. D. (1975). *Theory of optimal search*: Academic Press New York.

Stone, L. D. (1983). The process of search planning: current approaches and continuing problems. *Operations Research, 31*(2), 207-233.

CHAPTER THREE: PROBLEM STATEMENT

In this research we study dense storage environments that receive requests for items and these requests are fulfilled in a manual operating environment. Adding to the complexity of the studied problem, we assume that the exact locations of stored items are completely unknown to the order picker. In a dense storage environment, stored items may have to be moved around or repositioned to other open locations in order to gain access to other deeply stored items. Not knowing exact items' locations requires searching for the requested item and the search procedure requires considering what to do with the items that must be moved out of the way.

As mentioned in the first two chapters, we are motivated by the U.S. Navy concept of Sea Basing. In addition to the Sea Basing concept, the two major bodies of literature on which our contribution is based were surveyed in these two chapters, i.e. dense storage and search theory.

The first research contribution concentrates on developing a search plan for a single item in a special class of manually operated dense storage environments that have an inverted T configuration. While in the second contribution, we extend our work by relaxing the assumption of retrieving one item at a time, and instead consider search plans for orders consisting of multiple items that are to be retrieved from a dense storeroom with s-pallets stacked over each other.

<u>Contribution 1: Search Plan for A Single Item in an Inverted T K-Deep Storage System</u>

In chapter 4, we study a search planning problem for a single item in an inverted T k-deep storage system where exact stored items' locations are completely unknown to the picker, and moving items out of the way is required during the search process. It is assumed that the storage system is a non-depleting one. A non-depleting storage system, is the storage system where the

order picker or searcher picks cases from the stored pallets, the thing that leaves the number of stored pallets and overall system configuration remain the same (Futcher, 2003).

The research objectives of this problem are:

- Define the inverted T storeroom entirety by its accessibility constant k and its length h and define a unique coordinate system in which each cell holding one pallet is defined by its x and y coordinates.
- Develop a manual search plan heuristic for a single item in an inverted T dense storage environment having a single searcher. When followed, the search plan results in a search sequence array Ψ, which lowers the expected search time under certain assumptions.
- Define a search time equation for an item in the environment under study and derive analytical models to calculate each component of that equation based on the proposed coordinate system.
- Define two repositioning policies. The first repositioning policy assumes items can be repositioned in open aisle locations in the storeroom grid. In such a policy, the items need to be put-back during the search procedure to prevent blocking the way to the input/output point. In the second policy items are repositioned outside the storeroom grid, which requires the availability of extra space, but has the advantage that put-back can be completed after and independent of the search procedure.
- Use the proposed search time equation to calculate the search time for each cell in the storeroom.
- Obtain the entire range of search times and the search time probability distribution for each storeroom with configuration and density defined by k and h.

- Calculate both actual and approximate expected search time and standard deviation for each storeroom configuration based on the derived probability distribution for some storeroom configurations.

Thereafter, we would be able to:

- Equip the order picker with an operational search plan in the storeroom that when followed will result in the search time distribution that can be calculated a priori.
- Derive the probability distribution of search time of items in a dense storeroom with inverted T configuration. The advantage of having the entire probability distribution, is that it provides the decision maker with the full range of potential outcomes, consequently, allowing the calculation of not only the expected search time, but also the probability that the search time will be less or greater than a certain value, or between two values.
- Decide on a certain inverted T dense storeroom configuration defined by k and h based on the maximum search time of an item in that storeroom or based on the mean or expected search time of the entire storeroom.

<u>Contribution 2: Search Plan for Multiple Items in an Inverted T Dense Storage System</u>

In this part of the research we extend our first contribution to consider that multiple or M items are requested for retrieval, and such items are to be retrieved from a dense storeroom with s-pallets stacked over each other. Therefore, the studied problem is a search planning problem for multiple M items in an inverted T k-deep, s-pallets stacked, storage system where exact stored items' locations are completely unknown to the picker or the searcher. It is assumed in this contribution as well that the storage system is a non-depleting one. The main objective

here is to develop a manual search plan heuristic for M items in an inverted T, stacked dense storage environment having a single searcher.

A few notes are worth mentioning about this second contribution:

- Using the available storage space efficiently and increasing the number of stored pallets can be achieved by stacking pallets of items over each other, whether these stacked pallets carry identical items or not.
- We are investigating the effect of searching and retrieving M items from the storeroom not just a single item and studying the effect of doing that on the expected search time and the standard deviation of search times, especially while having vertical stacks of pallets.

The research objectives of this problem are:

- Define a k-deep, s-stacked inverted T dense storeroom entirety by its accessibility constant k, its length h, and stack height s and define a unique coordinate system in which each cell holding one pallet is defined horizontally by the x and y coordinates and vertically by the z coordinate.
- Develop a manual search plan heuristic for multiple or M items in an inverted T dense storage environment having a single searcher. When followed, the search plan results in a search sequence array Ψ, which lowers the expected search time under certain assumptions.
- Redefine the search time equation components introduced in the first contribution to calculate the search time for each cell in the storeroom. The new set of equations accounts for pallets stacked over each other, i.e. the vertical height in the storeroom.

Additionally, a new set or equations is introduced to calculate the retrieval time of pallets from different locations in the storeroom to the Input/ Output location.

- Calculate the expected search time and standard deviation of search times for each storeroom configuration defined by *k*, *h*, *s* and *M*.
- Test the performance of the proposed search plan heuristic using a set of designed experiments and analyze the outcomes using Analysis of Variance (ANOVA).

Thereafter, we are able to:

- Generate a search plan for each storeroom configuration using the search plan heuristic.
- Decide on a certain *k*-deep, *s*-stacked inverted T dense storeroom configuration defined by *k*, *h*, and *s* based on the maximum search time of an item in that storeroom or based on the expected search time of the entire storeroom or even a certain level of variability between search times.
- Decide on a number of pallets to be requested from a certain storeroom based on maximum desired expected search time or maximum value of standard deviation.

CHAPTER FOUR: SEARCH PLAN FOR A SINGLE ITEM IN AN INVERTED T K-DEEP STORAGE SYSTEM

Abstract

We study the problem of searching for a single item in an inverted T k-deep dense storage systems with uncertainty of item locations using a single searcher. This is a challenging environment because to gain access to some items, other items may need to be moved out of the way. Thus, search heuristics are presented that consider how to conduct repositioning and put-back operations, in addition to traveling, within a search procedure. Two repositioning policies are studied. We find that a repositioning policy that uses the open aisle locations as temporary storage locations and requires put-back of these items while searching is recommended as it results in lower expected search time and lower variability than a policy that uses available space outside the storage area and handles put-back independently of the search process. Statistical analysis of both repositioning cases shows that the probability distribution of search times in a storeroom with small configurations in terms of the accessibility constant and length can be approximated using a lognormal probability distribution with a certain mean and standard deviation. Knowing the probability distribution provides the decision makers with the full range of all possible probabilities of search times, which is useful for downstream planning operations.

Introduction

The two primary objectives associated with the distribution of goods are to use storage space and labor efficiently (Bartholdi & Hackman, 2008). One way to improve storage density is through the use of dense storage systems (Gue, 2006). In a dense storage system, "pallets are stacked on the floor in a lane that could be several pallets deep, and several pallets high". In

dense storage systems, not all of the stored items are directly accessible; therefore, some items may need to be moved out of the way to gain access to other items. Consequently, there is a trade-off associated with storage density and labor efficiency, e.g. increasing storage density decreases labor efficiency.

Gu, Goetschalckx, and McGinnis (2007) classified warehouse operations into three distinctive areas: receiving and shipping, storage, and order picking. Frazelle (2002) pointed out that order picking is considered very labor extensive and time consuming. Existing warehousing literature assumes that the exact locations of stored items are known to the picker, which is not always true in practice. In this work we assume that information about exact items' locations in the warehouse is missing, which adds to the complexity of the existing environment by forcing the picker to first search for the requested item before the picker can retrieve it. We develop an operational search policy of a distribution warehouse. The addition of the searching component in the traditional routing problem in order picking within the warehousing literature is considered novel and up to our knowledge has not been looked into before. Searching in a dense storage environment is particularly challenging because it can require moving units out of the way during the search process.

Figure 4 shows different elements of the search procedure for an item in a dense storage environment. The red arrow represents the travel component during the search procedure, the blue arrow represents the repositioning effort carried out by the searcher in order to gain access to deeply stored items in such environment, and the orange arrow represents the put-back effort that might be carried out by the searcher at different points of time during the search sequence in order to put-back repositioned items to their old storage locations before continuing with the search procedure.

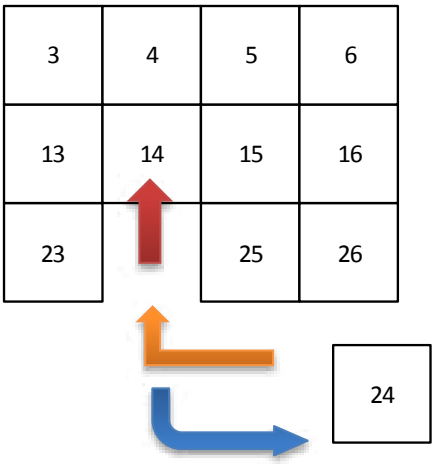

Figure 4: Components of the search procedure

We are motivated in this work by one of the US Navy's fundamental concepts for the 21st century, which is the Sea Basing concept. The main purpose behind having a sea base is to minimize the logistics footprint ashore while maintaining the capability of providing logistical sustainment of forces ashore participating in a variety of missions (Clark, 2002). Sea bases will act as floating distribution centers "providing ready issue material to forces ashore". These distribution centers will utilize the concept of dense storage, i.e. ships' storerooms are configured in a way that allows the maximization of available cargo onboard. However, the ships of the sea base must have selective offloading capability, which is the ability to locate and deliver any piece of cargo on board to support the forces ashore (Futcher, 2003). The concept of conducting operations on and from the sea has been given greater attention from both the US Navy and its allies abroad in order to be able to face common present and future threats. In 2006, the Combined Joint Operations from the Sea Center of Excellence (CJOS COE) was established with the membership of the US and 12 other allies. The main objective of CJOS COE is to investigate the possibilities of conducting safe and effective joint operations on or from the sea. A 2012 report by the CJOS COE reintroduced the sea basing concept and

widened its application to include both the US and its allies worldwide (CJOSCOE, 2012). The US Navy has made progress in understanding and designing the sea base concept, but the best way to conduct distribution and supply operations from a sea base is still an open challenge, as described in the 2013 article published in the *National Defense Magazine* about the problem of sea based supply lines and the effort exerted by the Marine Corps in order to reach the best possible way to "resupply combat troops from the sea" (Parsons, 2013).

We develop analytical search plan models to estimate search times of items in a dense storage system, given a specific configuration of the system, imperfect visibility of items' locations due to the absence of tracking and identification devices, and the use of manual order retrieval system. The need to search for an item in a dense storage environment cannot be entirely eliminated by the use of asset tracking devices such as RFID systems. For instance, asset tracking systems such as RFID cannot give the exact locations of every single item in a dense storeroom with large number of stored items, instead it can only be used to confirm whether a certain item is present in a certain storeroom or not. Additionally, the continuous movement and shifting of locations of large number of stored items resulting from the operation of the storage system causes uncertainty about items' locations to propagate through the entire system, such uncertainty cannot even be eliminated through the use of license plates such as the ones used in a traditional aisle-based storage system (Scala, 2015). Past research such as the work of Hariharan and Bukkapatnam (2009)studied the problem of searching in an aisle-based storage system with RFID tracking. An aisle-based storage system is a less complex system than the one studied here. Their research showed that, even the presence of an RFID system does not entirely eliminate the need for searching. Finally, asset visibility onboard of sea base ships is a strategic issue still being investigated by the Department of Defense (DoD). The DoD's strategy for improving asset visibility was published recently in 2014, and its main

objective is to "enhance the asset visibility in a manner that provides the ability to track assets throughout their lifecycle."; and one of the main approaches outlined in that strategy is to "build on existing infrastructure"(DoD, 2014).

The purpose of this research is four-fold:

1. Provide a first-time study of the search problem in a warehousing environment, specifically a dense storage environment. Retrieving items in a dense storage environment is challenging because items have to be moved out of the way in order to gain access to other deeply stored items. Therefore, the search process will have three components as shown in Figure 4: traveling in open aisles, repositioning items, and putting back items that have been repositioned in order to avoid impeding the way to the input/output point.
2. Develop a search plan heuristic for a single item in a dense storage environment with an inverted T configuration. An inverted T configuration is a configuration with one horizontal aisle and one vertical aisle as displayed in Figure 5. This configuration is recommended by Gue, 2006 because it is able to achieve high storage densities where at most k-1 items will be required to be moved out of the way to gain access to any item, where k is known as the accessibility constant. Two repositioning policies are studied and a search plan heuristic is developed for both of them. The first repositioning policy assumes items can be repositioned in open aisle locations in the storeroom grid. In such a policy, the items need to be put-back during the search procedure to prevent blocking the way to the input/output point. In the second policy items are repositioned outside the storeroom grid, which requires the availability of extra space, but has the advantage that put-back can be completed after and independent of the search

procedure. This repositioning policy is applicable in an environment where finding a certain item is given higher priority over other operations inside the storeroom.

3. Derive each component of the search time equation, and use it to calculate the search time for each item in the storeroom, as well as the expected search time and standard deviation for each storeroom configuration.
4. Provide insights about the design of searching in very high density storage systems with item location uncertainty through analyzing the results and propose operational search plans, which can be easily articulated to human searchers, for each of the two repositioning policies.

The remainder of this chapter is organized as follows: the next section surveys the literature on dense storage and search theory. After that we provide the problem formulation and the assumptions of the model. We provide an item search heuristic for inverted T storeroom configurations following the problem formulation. Then we present a summary of the numerical experiments and statistical analysis. And finally, we provide conclusions and suggestions for extending this research.

Literature Review

In terms of the surveyed literature, the problem under study lies on two pillars. The first pillar is the literature of dense storage environments and the second one is the literature of search theory, especially search plans in discrete environments.

Dense storage

Over the years engineers working in the warehousing and distribution industries have been developing ways to increase the storage density (Gue, 2006). Several publications, such as

Rana (1990), have discussed one way of increasing the storage density in a warehouse through dedicating less aisle space by having narrow or very narrow aisles. However, Parikh and Meller (2010) discussed some of the resulting problems of such systems, such as travel congestion or restricted travel in one-way aisles. A second way to increase the storage density is having k-deep pallet racks, where each location contains k pallets behind each other. Typically, such systems are designed such that each lane has the same *Stock Keeping Unit* (SKU), accordingly, no moving is required of one item to access the other. One form of the k-deep systems is the double-deep automated storage and retrieval systems discussed by Lerher, Sraml, Potrc, and Tollazzi (2010). Another form is through creating block stacking areas, in which pallets for a single SKU are stacked on the floor in a lane that could be several pallets deep and two or more pallets high. For a recent review of the block stacking models and the problems associated with such a method, the reader is referred to Matson, Sonnentag, White, and Imhoff (2014).

The most similar work to ours is the work on very high density storage systems. Gue (2006) mentioned that a storage system is said to have very high density (VHD) when "it is sometimes necessary to move interfering items in the system in order to gain access to desired items". In a k-deep very high density storage system, one might have to move up to $k - 1$ pallets in order to gain access to the desired pallet. For instance, in a 2-deep or double-deep system, at most one pallet has to be moved in order to retrieve the one behind it. Contrastingly, a single-deep storage system is not considered a VHD system, as all pallets are accessible directly from aisles. The letter k will be referred to in this work as the accessibility constant, which is according to Futcher (2003) "a variable directly related to the accessibility or physical access of any particular pallet" in a dense storage system. When the VHD storage system is automated, it is usually called a deep bulk storage system (Gue, 2006). In this work, we study manual dense

storage environments with inverted T configurations, which have one horizontal and one vertical aisle (Gue, 2006). An inverted T configuration provides the lowest mean retrieval times with a certain fixed value of the accessibility constant *k* Futcher (2003).

Search theory

Search theory originated from the work of Koopman (1956) and others during World War II. It is concerned with the problem of how to allocate a limited amount of resources to find a target (or targets) whose location is not precisely known, such that the reward, as specified by the measure of effectiveness, is maximized (Frost & Stone, 2001). Lau, Huang, and Dissanayake (2005) provide an extensive review of different applications of the search theory.

The problem we study here can be classified as a stationary target search problem. A stationary target search problem is the problem of finding a stationary target located at a point on a plane or alternatively in one of a number of discrete cells, and a search plan is to be developed and followed in order to find the target item with a certain objective, for example with the objective of minimizing the expected time to find that item (Lau, 2007; Lössner & Wegener, 1982). Specifically, the problem studied here can be classified as a discrete stationary target search problem, opposing to problems that fall under search planning in a continuous space and time with a moving target such as the work of Kierstead and DelBalzo (2003). The studied problem is classified as a discrete stationary target problem, because we are searching for a single stationary pallet among a set of pallets in a store room. However, one of the main differences between existing search theory problems and the studied problem is that while searching in a dense storage environment pallets have to be moved out of the way in order to gain access to another one.

Stone (1983) studied different elements of search planning. He presented an approach that encompasses both objective and subjective methods and provided compromises between theoretically optimal plans and operationally feasible ones. In this work we are interested in developing an operational search plan, because we are studying a manual environment and it is not always possible to provide human searchers with optimal search paths to follow.

A more recent work that deals with the idea of searching in a warehousing environment, is the work of Hariharan and Bukkapatnam (2009). They develop a search model based on partially observed Markov Decision Process to study the impact of using a RFID system that exhibits noise to locate a "misplaced item" in a warehouse. One of their results is that searching for a misplaced item in a warehouse with a poorly designed RFID system may "actually cost the firm more than without it". One of the main differences among this work and ours, is the environment under examination. While they are considering a traditional aisle based storage system, we are studying a dense storage system with an inverted T configuration, which requires considering the movements to reposition items to gain access to other items, while searching.

In this work we are studying an Inverted T dense storage system that is operated manually, and adding to the complexity of the problem, it is assumed that the exact locations of stored pallets are not known to the order picker, and a search procedure has to be followed inside the storeroom to find the item the picker is looking for. The search procedure in dense storage environments requires items to be moved out of the way during searching. This requires incorporating how to conduct repositioning and put-back movements in an operational search plan.

Problem Formulation

The problem under study is a search planning problem with the objective of minimizing the time required to detect the location of a certain item given imprecise knowledge of its exact location. The time required to detect the location of a certain item may require moving items out of the way during the search process; thus, the search time is the combination of travel, repositioning, and put-back times. The array Ψ represents the search sequence which when followed provides a low time to detect a certain item.

The following assumptions are made about the studied problem:

- When the search process starts, there is an equal probability of having the item stored in any location or cell.
- The number of searchers in the studied system is equal to one.
- Orders to retrieve a certain item are received at the input/output (I/O) location, which is assumed to be fixed and with known coordinates. Orders are received to retrieve a single item at a time.
- Search regions are defined as non-overlapping cells of equal sizes (i.e., one cell can hold one pallet). All cells' locations are known with certain coordinates.
- The searcher can guarantee a target item's presence or absence after the cell has been searched for a negligible amount of time and equal for all locations. Consequently, it is assumed that the searcher will be able to look at the identification tag on the pallet that provides its contents and quickly identify if it is the requested item or not.
- All items are one unit high in the vertical space direction, i.e. stacked storeroom configurations are not considered.

- Travel times are directly proportional to travel distances. Each unit distance is assumed to correspond to one cell distance and takes one unit time to be traveled for either unloaded or loaded trips.
- The studied system is assumed to be a non-depleting storage system (e.g., cases are picked off pallets), where the number of pallets present in the system remains the same.
- The storeroom inverted T configuration remains the same after each retrieval of an item.

In order to calculate the time required for searching analytically, a coordinate system has to be defined for the studied environment. The origin point ($x = 0$ and $y = 0$) is assumed to be located at the intersection between the vertical and horizontal aisles, and the I/O cell is located at $x_j = -k$ and $y_j = 0$. The value of the x coordinate increases to the right until it reaches the value of the accessibility constant k, and decreases in the opposite direction until it reaches the value $-k$. The value of the y coordinate decreases in the negative direction until the values of y reaches $-k$, and increases in the positive direction until the total number of pallets' rows becomes equal to h, where h is the storeroom length. To maintain the inverted T configuration, h has to be greater than or equal $2k + 2$. The two variables k and h are the only two parameters defining the storeroom configurations studied here.

Figure 5 shows a 5 by 6 2-deep inverted T storeroom with coordinates defined for each cell; consequently, $k = 2$ and $h = 6$.

3	1	2	3	4	5
2	6	7	8	9	10
1	11	12	13	14	15
0	-k = -2 (16)	-1 (17)	0 (18)	1 (19)	k = 2 (20)
-(k-1) = -1	21	22	23	24	25
-k = -2	26	27	28	29	30

Figure 5: Sample inverted T system with $\boldsymbol{k} = \mathbf{2}$ and $\boldsymbol{h} = 6$.

Any storeroom is divided into a number of N cells and stores a number n pallets in it. We define the N cells as follows. The set C is the set of cells indexed on $c = 1,2,\ldots,N$. The set C_O is the set of occupied cells, where, $c_o = 1,2,\ldots,n$ and c_o is a subset of C. Thus, each configuration's storage density can be calculated as $\frac{n}{N} = \frac{h(2k+1)-(2k+1)-(h-1-k)}{h(2k+1)}$. In Figure 5, the number of cells in the shown $5X6$ storeroom is $N = 30$, the number of occupied cells with pallets is $n = 22$, and the storage density is $\frac{22}{30} = 0.733$.

The set of cells C consists of the two disjoint subsets: the set of occupied cells C_O; and the set of unoccupied cells C_U. The set of occupied cells includes five disjoint subsets; $C_O = C_O^{UL} \cup C_O^{UR} \cup C_O^{BL} \cup C_O^{BR} \cup C_O^{BC}$, where, C_O^{UL} is the set of occupied cells in the upper left part of the storeroom, C_O^{UR} is the set of occupied cells in the upper right part of the storeroom, C_O^{BL} is the set of cells in the bottom left part of the storeroom, C_O^{BR} is the set of cells in the bottom right part of the storeroom, and C_O^{BC} is the set of cells in the bottom center part of the storeroom location at $x = 0$ and from $y = -1$ to $y = -k$.

It is assumed that pallets located at any cell $j \in C_O^{UL} \cup C_O^{UR}$ are accessed only through the vertical aisle and are always repositioned to the bottom right horizontal aisle. Therefore, pallets

with y-coordinate $y_j = 1$ are assumed not to be repositioned to gain access to other deeply stored pallets through the horizontal aisle. The main reason for this assumption is that the number of pallets above the horizontal aisle will always be greater than or equal to the number of pallets that are to the left (or right) of the vertical aisle. Also, this assumption results in each occupied cell having a single nearest accessible open aisle location. Similarly, it is assumed that pallets located below the horizontal aisle (i.e., at any cell $\in C_O^{BL} \cup C_O^{BR} \cup C_O^{BC}$) are accessed only through the horizontal aisle and are always repositioned to the open locations in the vertical aisle.

The search plan problem is represented in a network form. The nodes of the network represent the cells being searched, and the weights of the arcs represent the summation of the time needed to travel between a pair of cells, plus the time needed to reposition pallets blocking the searcher's access to a certain cell (if any), plus the put-back time of already repositioned pallets to their storage location before traveling to cell j coming from cell i (if any). The weight between cells i with coordinates x_i and y_i and cell j with coordinates x_j and y_j is denoted as $T_{(x_i,y_i),(x_j,y_j)}$ and calculated as follows:

$$T_{(x_i,y_i),(x_j,y_j)} = \tau_{(x_u,y_u),(x_v,y_v)} + R_{(x_j,y_j)} + P_{b(x_i,y_i),(x_j,y_j)} \quad (1)$$

Where, the travel time is denoted as $\tau_{(x_u,y_u),(x_v,y_v)}$ between cell i with coordinates x_i and y_i and cell j with coordinates x_j and y_j through open aisle locations u and v with coordinates (x_u, y_u) and (x_v, y_v), respectively. $R_{(x_j,y_j)}$ denotes the repositioning time to gain access to the pallet located at cell j. And $P_{b(x_i,y_i),(x_j,y_j)}$ is the put-back time of already repositioned pallets to their storage location before traveling to cell j coming from cell i. The three time components will be discussed in details in the following subsections.

As the searcher progresses with his search for the target item in the storeroom, the probability of finding the item in a certain cell j changes, either it becomes equal to 1, i.e. the target item is located in cell j, or becomes equal to zero, i.e. the target item is not in cell j. Therefore, when making a decision about the next cell to search, only unsearched cells are considered.

Travel Time

Travel time is defined as the time taken by the searcher to travel through the aisle until he reaches a certain cell. This cell can be the one that faces the targeted item, or it can be the cell from which the searcher will start the pallet repositioning activity to move pallets blocking his way in an attempt to find the target item. Therefore, travel time measures the distance traveled through unoccupied cells.

The travel time from cell i to cell j (τ_{ij}) for every i and j belonging to the set of unoccupied cell C_U is directly proportional to the rectangular distance between i and j. This distance is equal to $\tau_{ij} = |x_j - x_i| + |y_i - y_i| \; \forall i \in C_U \; and \; j \in C_U$.

If cells i and j belong to the set of occupied cells C_O, travelling still occurs through aisles or unoccupied cells. For instance, if the searcher wants to go from occupied cell $i = 26$ to occupied cell $j = 1$, traveling occurs from unoccupied cell $u = 16$ to unoccupied cell $v = 3$.

Therefore, for each cell i belonging to the set of occupied cells C_O, there is a unique cell u that belongs to the set of unoccupied cells C_U, where u represents the closest accessible aisle location to i. The determination of the nearest unoccupied cell to a certain cell i that belongs to the set of occupied cells C_O depends on the location of cell i in the storeroom and thus can be determined a priori.

If any cell i with coordinates $\{x_i, y_i\}$ belongs to the set of upper occupied cells C_O^U <u>and $|y_i| > k$</u>, they will be accessed using the vertical aisle and the nearest open aisle location u is located at $x_u = 0$ and $y_u = y_i$. For example, the nearest open location u with coordinates $\{x_u, y_u\}$ to cell $i = 1$ $\{-2, 3\}$ is located at $x_u = 0$ and $y_u = y_1 = 3$. If any cell i belongs to the set of upper occupied cells C_O^U <u>and $|y_i| < k$</u>, the nearest open aisle location u is located at $x_i = x_u$ and $y_u = 0$.

If any cell i belongs to the set of bottom occupied cells C_O^B, they will be accessed using the horizontal aisle and the nearest open aisle location u is located at $x_i = x_u$ and $y_i = 0$.

An example using the storeroom configuration shown in Figure 5 is provided that calculates the travel time between cell 23 and cell 4. The first step is to check which set of cells each of cell 23 and cell 4 belong to. Both cells 23 and 4 belong to the set of occupied cells C_O. Cell 23 belongs to the set of bottom occupied cells C_O^B, and cell 4 belongs to the set of upper occupied cells C_O^U with <u>$|y_4| > k.$</u> The second step is to determine the corresponding unoccupied cell for each of cell 23 and cell 4. Since cell 23 belongs to the set of bottom occupied cells C_O^B, therefore, the nearest open aisle location u is located at $x_u = x_{23} = 0$ and $y_u = 0$, i.e. cell 18 $\{0, 0$. Since cell 4 belongs to the set of upper occupied cell C_O^U with <u>$|y_4| > k1,$</u> therefore, the nearest open aisle location is located at $x_u = 0$ and $y_u = y_4 = 3$, i.e. cell 3 $\{0, 3\}$. Finally, calculate the travel time distance between cells 18 and 3 as shown in Equation 2.

$$\tau_{18,3} = |x_3 - x_{18}| + |y_3 - y_{18}| = |0 - 0| + |3 - 0| = 3 \tag{2}$$

Repositioning Time

Repositioning time (R_j) is the time needed by the searcher to move a pallet out of the way in order to gain access to pallet j located at coordinates x_j and y_j. Two policies will be

investigated here, the first repositioning policy models an environment where pallets are temporarily repositioned inside the storeroom to an unoccupied cell in a manner that will not impede the searcher's future path to the I/O location. The second policy models an environment where pallets are repositioned outside the storeroom grid, which requires that there is available space outside the storeroom.

Policy 1: Repositioning occurs inside the storeroom

In this policy, pallets are repositioned inside the storeroom to unoccupied open cells, where they are repositioned in a way that will not impeded the searcher's way to I/O location after finding the item he was searching for.

Before deriving equations to calculate the repositioning time in policy 1, we need to define a repositioning rule. Following the repositioning rule proposed by Futcher (2003), the repositioning rule applied can be summarized as follows:

1. Reposition all the pallets that block the target pallet's j way to the I/O location in the open aisle locations that do not impede the target pallet's path to the I/O point.
2. Repositioned pallets are moved to the first unoccupied cells not along the exit path to the I/O point.

Generally, the proposed repositioning rule is characterized by the following:

- The maximum number of pallets to be repositioned in order to gain access to the most deeply stored pallet is equal to $k - 1$, where k is the accessibility constant.
- The blocking pallets that need repositioning are moved to the closest empty aisle locations that do not impede the path of the pallet that we need to gain access to if it is indeed the target pallet. This means pallets located at any cell $j \in C_O^{UL} \cup C_O^{UR}$ will be

repositioned to an unoccupied cell location on the horizontal aisle above cells C_O^{BR}, and pallets located at any cell $j \in C_O^{BL} \cup C_O^{BC} \cup C_O^{BR}$ will be repositioned to an unoccupied cell location on the vertical aisle across from C_O^{UL} and C_O^{UR}.

Repositioning distance, in order to gain access to the certain pallet in cell *j,* and consequently repositioning time, depends in its calculation on the position of the pallet *j* that needs to be retrieved. In order to develop repositioning distances equations that can be calculated a priori, the repositioning distances represent a worst-case expression because they are based on repositioning the item to the farthest open aisle location based on their depth. For example, repositioning items in the upper cell locations that have y-coordinates of +/-1 are assumed to be repositioned to the unoccupied cell in the horizontal aisle with y-coordinate of k. This allows the opportunity for additional items to be repositioned in the aisle, but sometimes overestimates the repositioning (for example, if both items in cell 2 and cell 7 in Figure 5 are repositioned, one would be repositioned to cell 20 and another to cell 19), however, the equations calculate the repositioning distance based on moving them both to cell 20.

The following equations give the repositioning distances of individual pallets that block the way of the target pallet that needs to be retrieved as a function of the location of that pallet *j*. As a reminder, these repositioning distance equations are based on the assumption that the searcher's access to deeply stored pallets stored above the horizontal aisle of the storeroom can only be accessed through the vertical aisle, and access to deeply stored pallets stored below the horizontal aisle are accessed through the horizontal aisle.

For any pallet located at cell j $\{x_j, y_j\}$ belonging to the set C_O^{UL} of occupied cells in the upper left part of the storeroom, or belonging to the set C_O^{UR} of occupied cells in the upper right part of the storeroom, or belong to the set C_O^{BR} of occupied cells in the bottom right part of the

storeroom the total repositioning time required to gain access to this pallet is directly proportional to the total repositioning distance, which is equal to

$$R_{(x_j,y_j)} = 2 \times [1 + |y_j| + |x_j|] \tag{3}$$

Where, 2 is the multiplicative factor that accounts for the searcher's trip from repositioned pallet's repositioning location to its original storage location; and x_j and y_j are the x and y coordinates of pallet located at cell j.

For any pallet located at cell j $\{x_j, y_j\}$ belonging to the set C_O^{BL} of occupied cells in the bottom left part of the storeroom, the total repositioning time required to gain access to this pallet is directly proportional to the total repositioning distance, which is equal to

$$R_{(x_j,y_j)} = 2 \times [|y_j| + |x_j|] \tag{4}$$

For any pallet located at cell j $\{x_j, y_j\}$ belonging to the set C_O^{BC} of occupied cells in the bottom center part of the storeroom, the total repositioning time required to gain access to this pallet is directly proportional to the total repositioning distance, which is equal to

$$R_{(x_j,y_j)} = 2 \times |y_j| \tag{5}$$

As an example, the repositioning time of the pallet in cell 2 in order to gain access to pallet 1 in the system shown in Figure 5 is calculated as follows. Pallet 1 belongs to the set C_O^{UL} of upper left occupied cells and the pallet in cell 2 will be temporary repositioned to cell 20; therefore, the repositioning distance is:

$$R_{(x_1,y_1)} = 2 \times [1 + |y_1| + |x_1|] = 2 \times [1 + |3| + |-2|] = 12 \tag{6}$$

Policy 2: Repositioning occurs outside the storeroom

In this policy, pallets that need repositioning are repositioned outside the storeroom grid to a fixed location. This repositioning policy represents an environment where the repositioned pallets are left outside the storeroom until the item we are looking for is found in the main storeroom and retrieved to the I/O location. This fixed repositioning location is assumed to have coordinates $x = -k - 1$ and $y = 0$ as shown in Figure 6.

3		1	2	3	4	5
2		6	7	8	9	10
1		11	12	13	14	15
0	-k-1 = -3 repositioning	-k = -2 (16)	-1 (17)	0 (18)	1 (19)	k = 2 (20)
-(k-1) = -1		21	22	23	24	25
-k = -2		26	27	28	29	30

Figure 6: Repositioning outside the storeroom grid

The main motivation of repositioning pallets outside the storeroom grid is to avoid the need to put-back repositioned pallets to their old storage locations while the search procedure is being carried out. Instead, putting back of repositioned pallets temporarily stored outside the storeroom will be done after finding the item the searcher is looking for; thus, put-back is done independently of the search process. A motivation for considering this case was to analyze an environment motivated by a decision maker who suggested that finding a certain item we are looking for is given priority over other operations inside the storeroom. i.e. "I don't care if you have to remove everything, I just need you to find the requested item – now!" This case requires that there is available space outside the storeroom to temporarily store repositioned

pallets and available underutilized labor to put the pallets back after the search process is completed. As pallets are repositioned to a certain fixed location, the repositioning distance equations are different from those in repositioning policy 1. For any pallet located at cell $j\ \{x_j, y_j\}$ belonging to the sets $C_O^{BR}, C_O^{BC}, C_O^{BL}$ and C_O^{UR}, the repositioning distance is given by

$$R_{(x_j,y_j)} = 2\ \times \left[|y_j| + |x_j + k + 1| - 1\right] \tag{7}$$

For any pallet located at cell $j\ \{x_j, y_j\}$ belonging to the set C_O^{UL}, the repositioning distance is given by

$$R_{(x_j,y_j)} = 2\ \times \left[y_j + |x_j| + |-k - 1| - 1\right] \tag{8}$$

Repositioning Procedure

One critical aspect of the repositioning procedure is not to repositioned items that have direct contact with open aisle locations. For example, in Figure 5, pallet 11 seems to be deeply stored with respect to pallet 12; however, the searcher can access pallet 11 without the need to reposition pallet 12. Therefore, a repositioning subroutine is defined in order to be able to deal with such cases. This subroutine keeps track of whether the wth pallet searched requires repositioning through a repositioning pallets array *Repo*.

<u>Subroutine Reposition</u>

If (cell $j^* \in C_o^{UL}$) AND $(x_{j^*} \leq -2)$

Then start repositioning, set $Repo[w] = 1$

Else No repositioning needed, set $Repo[w] = 0$

If (cell $j^* \in C_o^{UR}$) AND $(x_{j^*} \geq 2)$

Then start repositioning, update $Repo[w] = 1$

Else No repositioning needed, set $Repo[w] = 0$

If $(i \in C_o^B)$ AND $(y_{j^*} \leq -2)$

Then start repositioning, set $Repo[w] = 1$

Else No repositioning needed, set $Repo[w] = 0$

Put-back Time

Put-back time (P_{bj}) is defined as the time needed to move repositioned pallets back to their storage locations before searching cell j. Again, we have two cases, case 1 in which repositioning of pallets is done inside the storeroom as previously defined, and case 2 where pallets are repositioned outside the storeroom. Put-back time is considered as part of the search process only in the first case, as put-back of repositioned pallets has to be done at certain points of time during the search procedure in order to prevent accumulation of pallets in open aisle locations to the point that the searcher's path to the I/O location will be impeded upon retrieval of the requested pallet. In the second case, the put-back effort is assumed to occur after and independently of the search process.

For *case 1*, the proposed put-back rule is characterized by the following:

- Repositioned pallets are put-back to the storage locations before moving to the next cell if the number of repositioned pallets is equal to the number of maximum allowed pallets to be repositioned, which is equal to the accessibility constant k.

The above rule is established because any further repositioning before putting back the existing repositioned pallets will impede the path of any pallet being retrieved to the I/O location. Priority is given to retrieve the target pallet in case it was found before put-back any of the repositioned pallets to their storage locations. The decision whether to initiate a put-back or not is made when the number of repositioned pallets reaches k and before traveling to the next

cell in the search procedure. Therefore, put-back distances are calculated using the same set of equations under *case 1* used to calculate the repositioning distances. Knowing these characteristics, the put-back procedure can be summarized using the following subroutine *Put-back*, which counts the number of repositioned pallets that haven't already been put-back using the *Repo* array.

<u>Subroutine Put-back</u>

If $\sum_{l=\emptyset}^{w} Repo[l] = \alpha k$

Then start put-back, and set $\alpha = \alpha + 1$

Else no Put-Back required

<u>Search Plan Heuristic</u>

The search plan heuristic starts by generating arcs from the current cell (initially, when a new order is received, the current cell is the I/O cell) and evaluating the weights (function of travel, repositioning, and put-back times) of each of those arcs. The cell connected with the arc with the least weight is selected. The indices of the cell connected with the arc with the least weight are added to the array Ψ of visited cells. This procedure is repeated until the target item has been detected. Having the sequence Ψ with the indices of visited cells and their coordinates, one can calculated the time taken to find a specific item in the storeroom.

The search plan heuristic for case 1, where pallets are repositioned inside the storeroom can be summarized through the following pseudo code:

1. Given: k and h.
2. Generate x and y coordinates for each cell $i \in C$, and classify each cell $i \in C$ in its appropriate subset ($C_O^{UL}, C_O^{UR}, C_O^{BL}, C_O^{BR}, C_O^{BC}, C_U$).
3. Determine each occupied cell $i \in C_O$ open aisle location u.
    a. If ($i \in C_o^U$ and $|y_i| \neq 1$) Then $x_u = 0$ and $y_u = y_i$
    b. If ($i \in C_o^U$ and $|y_i| = 1$) Then $x_u = x_i$ and $y_u = 0$
    c. If ($i \in C_o^B$) Then $x_u = x_i$ and $y_u = 0$

4. Start with an empty visit sequence array Ψ, and empty set of visited cells array ψ. Set $w = 0$ and $\alpha = 1$.
5. Start the search procedure at I/O cell, i.e. $\Psi[w] = I/O$ cell location.
6. Set $i = I/O\ cell$.
7. Calculate the distance $T_{(x_i,y_i),(x_j,y_j)} = \tau_{(x_u,y_u),(x_v,y_v)} + R_{(x_j,y_j)} + P_{b(x_i,y_i),(x_j,y_j)}$

   Between cell i and cell j $\forall\ j \notin \psi$ using travel, repositioning and put-back equations for *case 1* as shown above.

8. Select cell $j^*$with min T_{ij}; break ties arbitrarily.
9. Update $\Psi[w] = j^*$ and $\psi = \psi + j^*$
10. Travel from the nearest open aisle location of cell i to the nearest open aisle location of cell j^*
11. Do Subroutine *Reposition*
12. Do Subroutine *Put-back*
13. If $(ItemNumber = \Psi[w])$

    Then stop *search*

    Output: $\Psi[w]$.

    Else

    $w = w + 1, i = j^*$and go to step 7

Pseudo code for *case 2* follows a similar search procedure that modifies step 7 by using the appropriate equations to calculate the time components. Also, no put-back is required, thus step 12 can be ignored.

Operational Search Plan

Applying the search heuristic given in the previous section will generate the search array Ψ, which provides the sequence of cells to be visited. Because the target is stationary and the probability the target item is in any occupied cell is uniform, the cells to be visited in the search sequence do not change based on when in the sequence the target is found.

Thus, the search array Ψ is used to provide an operational search plan, which could be easily articulated to a human searcher, as follows: the search array Ψ recommends that the searcher visits all cells that can be accessed through open aisle locations before he needs to start

repositioning pallets to other open aisle locations in order to gain access to a deeply-stored pallet.

For the system shown in Figure 5, for example, assuming a *case 1* system where repositioning is done inside the grid, the search procedure starts after the searcher receives an order at the I/O cell located at cell 16 $\{-2, 0\}$. The searcher then follows the search sequence given by the search array:

$$\Psi_{case\ 1} = \{16, 11, 21, 12, 22, 23, 14, 24, 15, 25, 7, 9, 2, 4, 28, 29, 27, 26, 30, 6, 5, 1, 10\}$$

This search array follows the operational search plan, because the search procedure starts with visiting cells that have access to open aisles from beginning at cell 11 to cell 4. Starting from cell 28 in Ψ, repositioning is needed in order to gain access to deeply-stored pallets. For instance, in order to gain access to cell 28, the pallet located in cell 23 needs to be repositioned first. The cells visited follow an organ-pipe structure (BENGÜ, 1995). Put-back appears as well after visiting cell 29 and before traveling to cell 27, as gaining access to the deeply-stored pallet located in cell 27 needs the repositioning of pallet 22, making the number or repositioned pallets to increase to $k + 1 = 3$, which is not acceptable per the proposed put-back policy.

Although the specific search sequence Ψ will differ if a *case 2 repositioning policy* is applied because the repositioning equations are different, the recommended operational search plan still holds. For example, assuming that repositioning is done outside the gird for the system shown in Figure 6, the search sequence will be:

$$\Psi_{case\ 2} = \{16, 11, 21, 12, 22, 23, 14, 24, 15, 25, 7, 9, 2, 4, 26, 27, 28, 29, 10, 30, 6, 1, 5\}$$

The main difference between the two search sequences appears after visiting all cells that are in contact with open aisle locations, i.e. after visiting cell 4, the searcher then starts visiting

deeply-stored cells with low values of repositioning distance relative to the fixed repositioning location with coordinates $x = -k - 1$ and $y = 0$.

What remains now after knowing the search sequence, which can be translated into an operational search plan, is to calculate the search time for each cell in a certain configuration and to calculate the expected search time for the entire system and the corresponding standard deviation.

For a search sequence array $\Psi[w]$ with $w = [0, 1, 2, \dots, n]$. The search time for any cell $i \in w$, is calculated by:

$$T_i = T_0 + \sum_{j=1}^{i} T_{j-1,j} \; \forall i = 0, 1, \dots, n \tag{9}$$

Where, T_i is the search time for cell i in the search sequence $\Psi[w]$, T_0 is the search time for the first cell in the sequence $\Psi[w]$, $T_{j-1,j}$ is the search time between cell with index $j - 1$ and cell with index j in the array $\Psi[w]$. The expected search time, denoted as E, is calculated by:

$$E = \sum_{i=0}^{n} T_i / n \tag{10}$$

The standard deviation, denoted as SD, is calculated by:

$$SD = \sqrt{\frac{\sum_{i=0}^{n} (T_i - E)^2}{n - 1}} \tag{11}$$

Numerical Experiments

In this section we conduct numerical experiments to study the proposed search plan with all its components outlined in the previous two sections. A set of problems is generated, which are solved using the proposed search plan heuristics. The heuristics provide a search sequence, which is used to calculate three things for each of the generated problem instances: (1) the

search time to find an item in any given cell of the storeroom, (2) the expected search time for each storeroom configuration, and (3) the standard deviation of search times for each storeroom configuration.

Each problem instance represents a different storeroom configuration and is defined entirely by the accessibility constant k and the storeroom length h. In order to maintain the inverted T configuration of the storeroom, the minimum value of h has to be greater than $2k + 1$. In the experiments, a set of storeroom configurations was generated for each integer value between the following bounds: the accessibility constant k took the values between 2 and 20, and the values of h ranged between $2k + 2$ and 100 for each value of k. The search time for every possible cell in the storeroom is calculated using the proposed heuristic, and for each storeroom configuration the expected search time and standard deviation of search times were computed.

The algorithm is programmed in Python 3.4, and all experiments are conducted on a desktop computer supported by an Intel Core 2 Duo processor of 3.0 GHz CPU speed.

Figure 7 exhibits the search time for each observation in a storeroom configurations with $k = 2$ and $h = 10$ for repositioning policy 1. The observations are presented from left to right in increasing order based on search time. Thus, the first observation is the first cell in the search sequence and on the y-axis is the search time for that cell, the second observation is the second cell in the search sequence, and so on.

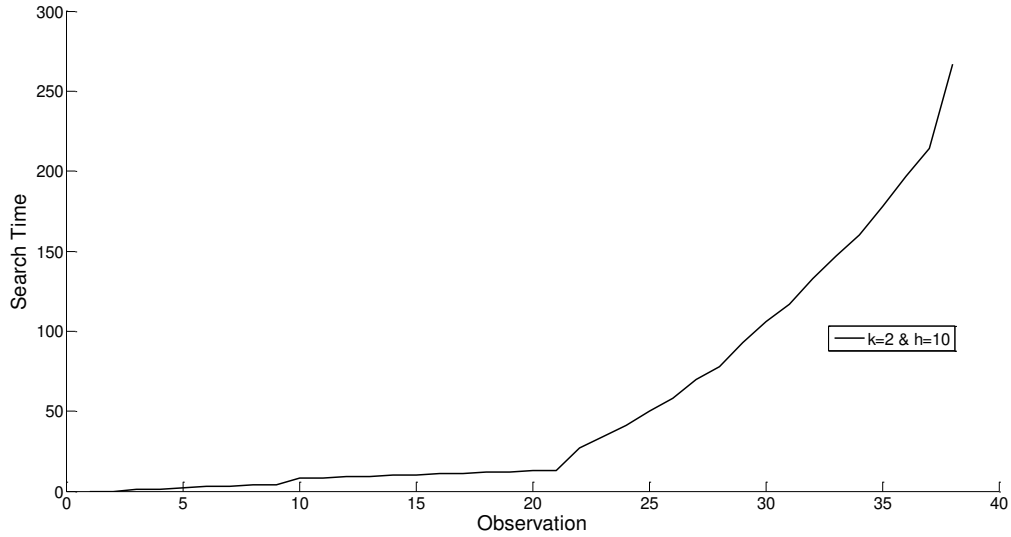

Figure 7: Search Time versus the Number of Pallets for a storeroom with $\boldsymbol{k} = \mathbf{2}$ and $\boldsymbol{h} = \mathbf{10}$ for repositioning policy 1

The curve in Figure 7 exhibits an increase in its slope after a certain portion of the curve almost parallel to the x-axis, followed by a second increase in the steepness of the slope of the curve. This first portion of each curve represents the locations that require only in-the-aisle travel to search, where all pallets searched at the beginning of the search procedure are directly accessible through open aisles and no repositioning is needed. The second portion represents the locations that require repositioning of pallets in order to gain access to deeply stored pallets occur. The third and final increase in the steepness of the curve represents the portion of the search procedure where the searcher needs to put-back repositioned pallets to their old locations before proceeding in the search procedure and moving to the next pallet. All other storeroom configurations, with different k and h values, that use repositioning policy 1 behave in the same three-tiered way.

Figure 8 shows the search time for each observation in a storeroom configurations with $k = 2$ and $h = 10$ for repositioning policy *2* ordered according to the search sequence. In the second

policy the sudden increase in the search time towards the end of the search cycle due to the addition of put-back times does not occur as it does in policy 1 as shown in Figure 8, as pallets are repositioned to a location outside the storeroom grid assuming the existence of extra space next to the main storeroom and assuming that the put-back will be done after finding the items the searcher was looking for.

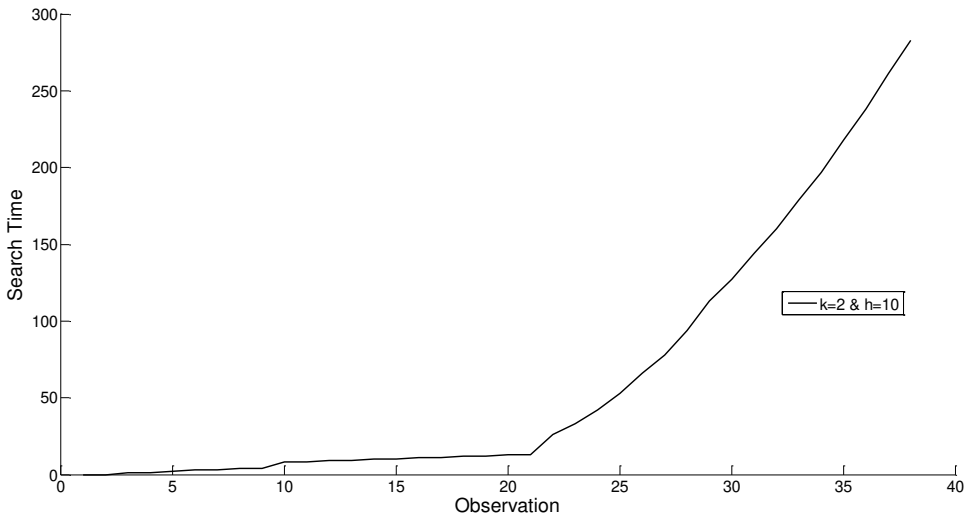

Figure 8: Search Time versus the Number of Pallets for a storeroom with $\boldsymbol{k} = \mathbf{2}$ and $\boldsymbol{h} = \mathbf{10}$ for policy 2

To summarize the experiments, in Table 3 we display configurations that vary the accessibility constants at *k = 2, 5, 10, and 20* and length values set at a short, medium, and tall length. For repositioning policies 1 and 2, the table provides the values of *k, h* and the corresponding values of expected search time and standard deviation of search times in time units, with each time unit corresponding to one unit distance. From the results displayed in Table 3, one can notice that both expected search times and standard deviations of the search time for each configuration are larger in values in repositioning policy 2, where pallets are repositioned outside the storeroom grid. This is especially interesting because repositioning policy 2 results do not include put-back efforts, whereas policy 1 does. As the accessibility constant and length

of the storeroom increase the values of expected search time and standard deviation exhibit a greater increase in repositioning policy 2. This behavior can be attributed to the fact that the searcher in policy 2 has to move longer distances and spend more time while he moves all pallets needing repositioning to the repositioning location outside the storeroom grid, on the other hand, in a policy 1 system, pallets are repositioned to the nearest possible repositioning locations. Therefore, an insight from our work is that a repositioning policy that uses the open aisle locations as temporary storage locations and requires put-back of these items while searching is recommended over a policy that uses available space outside the storage area and handles put-back independently of the search process.

Table 3: Expected Search Time and Standard Deviation of Search Times in Time Units for repositioning policies 1 and 2 with different *k* and *h* values

			Repositioning Policy 1 (inside storeroom)		**Repositioning Policy 2 (outside storeroom)**	
k	***h***	**Density**	**Expected Search Time**	**Standard Deviation**	**Expected Search Time**	**Standard Deviation**
2	6	0.733	27.5	35.4	27.8	35.9
2	50	0.792	931.9	1401.8	1003.3	1495.0
2	100	0.796	3515.8	5477.8	3674.3	5677.5
5	12	0.871	444.4	425.3	587.1	586.9
5	50	0.900	5432.5	5836.6	6574.9	6838.9
5	100	0.905	21293.5	23496.4	23780.5	25569.8
10	22	0.931	3546.5	3020.6	5176.4	4526.5
10	50	0.943	13778.8	12630.8	19187.4	17040.6
10	100	0.948	52176.0	51162.5	64735.0	60530.6
20	42	0.964	27319.2	21844.4	43439.7	34998.0
20	100	0.971	116845.8	102707.9	168959.1	140933.4

A few things can be noticed from the conducted experiments. As the value of the accessibility constant k increases, the density increases. While increasing h, for a constant value of k, also increases the storage density, it has less of an effect, and the value of storage density ranges

within a certain interval. For instance, when $k = 2$, varying h from 6 to 100 results in a storage density in the range between 0.733 and 0.796.

The Search Time function has three terms, the first term calculates the travel time, the second calculates the repositioning time and the third term calculates the put-back time. For storeroom configurations with lower densities and accordingly lower number of deeply stored pallets, the travel time component of the search time formula contributes the most to the value of the search time. As the density increases, the number of deeply stored items increases and accordingly the total search time increases rapidly as a large number of pallets will require repositioning to gain access to deeply stored pallets as well as the time needed to put-back some of repositioned pallets to their storage locations.

Another insight is that the values of the standard deviation, such as those displayed in Table 3, are large, with many having standard deviations greater than the values of their corresponding expected search time. Therefore, there exists a large amount of variability in the values of search times between one cell to the other in any storeroom configuration. In the dense storage systems studied here, we assume not knowing the exact locations of items in the storeroom, therefore, the picker has to search for a specific item before retrieving it, which reveals the cumulative nature of searching in a dense storage environment. The cumulative nature of searching results in the high range of search times, for a given storeroom configuration, because the last cell to search required repositioning and put-backs associated with cells earlier in the search sequence. Some items are found quickly, because they are early in the search sequence and accessible with no need to reposition or put-back any other pallets before being found, while others require cumulative effort in order to be found. Such extreme variability is in contrast to the case of traditional order picking in a dense storage environment. In retrieval

with item location certainty, some locations will be more convenient than others, however, the cumulative nature is not seen. Thus, for any given configuration, the variability of retrieval times will be much lower than the variability of search times.

One might be interested in determining if the search time distribution of a given configuration can be estimated using a known probability distribution. Therefore, we test the hypothesis that a certain theoretical distribution fits the search time data points for each storeroom configuration for both policies 1 and 2.

Hypothesis:

$$H_0: a\ certian\ theoretical\ distribution\ fits\ the\ given\ search\ time\ data$$

$$H_1: distribution\ does\ not\ fit\ the\ given\ data$$

The tested theoretical distributions are Exponential, Lognormal and Normal. The hypothesis test is carried at a level of significance $\alpha = 0.05$. The hypothesis testing is carried out using the Anderson-Darling (A-D) test, the A-D test statistic is calculated, and the $p - value$ for each of the four continuous distributions is compared against the specified level of significance $\alpha = 0.05$. When the $p - value$ is greater than α, it can be concluded that a certain probability distribution fits the given data. For more information about the Anderson-Darling test, the reader is referred to Law (2007).

As an example of the hypothesis test carried out, the A-D test for the storeroom with $k = 2$ and $h = 6$ for both policy 1 and policy 2. Table 4 summarizes the findings of the A-D test for each of the three continuous distributions for both repositioning policies with this system configuration.

Table 4: A-D test for storeroom with $\boldsymbol{k} = \mathbf{2}$ and $\boldsymbol{h} = \mathbf{6}$

	Repositioning Policy 1 (inside storeroom)		**Repositioning Policy 2 (outside storeroom)**	
Distribution	$\boldsymbol{p - value}$	**Result**	$\boldsymbol{p - value}$	**Result**
Normal	$0.005 < 0.05$	Reject H_0	$0.005 < 0.05$	Reject H_0
Lognormal	$0.396 > 0.05$	Do Not Reject H_0	$0.402 > 0.05$	Do Not Reject H_0
Exponential	$0.012 < 0.05$	Reject H_0	$0.009 < 0.05$	Reject H_0

Therefore, a lognormal distribution fits the search time data for both repositioning policies in storeroom configuration with $k = 2$ and $h = 6$. The lognormal probability distribution of the policy 1 system has a mean $\mu = 2.48979$ and standard deviation $= 1.53724$ and has a mean $\mu = 2.49275$ and standard deviation $= 1.54383$ for the policy 2 system with the same k and h.

The goodness-of-fit test was carried out on Minitab (2010) and a similar test procedure was carried out on the different configurations for both repositioning policies. However, as the values of both k and h increased, the lognormal probability distribution and other theoretical probability distributions did not provide a good for the search times. Table 5 shows the configurations that the lognormal probability distribution provided a statistically good fit for its search time values. The largest storeroom configuration that had probability distribution of its search times with an approximate lognormal distribution at a significant level of 0.05 is the storeroom with $k = 2$ and $h = 9$ for both repositioning policies and with $k = 2$ and $h = 10$ for repositioning policy 1. Bigger systems with either h greater than 10 or bigger values of k, did not pass the hypothesis test at a level of significance of 0.05. Thus, in subsequent discussion a small configuration is defined as one with $k = 2$ and h less than 11.

Table 5: Mean $\boldsymbol{\mu}$ and standard deviation $\boldsymbol{\sigma}$ for systems with search time probability distribution fitting a lognormal distribution

		Repositioning Policy 1 (inside storeroom)		**Repositioning Policy 2 (outside storeroom)**	
$\boldsymbol{k}$	$\boldsymbol{h}$	$\boldsymbol{\mu}$	$\boldsymbol{\sigma}$	$\boldsymbol{\mu}$	$\boldsymbol{\sigma}$
2	7	2.65370	1.52279	2.70116	1.58280
2	8	2.83268	1.56802	2.87352	1.61569
2	9	2.99821	1.61554	3.02112	1.64523
2	10	3.08786	1.60027	Does not fit	
2	11	Does not fit		Does not fit	
2	$h > 11$	Does not fit		Does not fit	
$k > 2$	$h > 0$	Does not fit		Does not fit	

Knowing that lognormal distribution is a good fit for the search time data for small storeroom configurations in both repositioning cases, we can estimate the probability that search time in a certain storeroom will be less than a certain value, or between two values, or greater than a certain value. By providing the decision maker with the distribution of search times for a certain storeroom configuration in terms of values of k and h , the decision maker has the full probability distribution of potential search times available to them. A decision maker can use the search time distribution to plan downstream activities, ensuring, for example, that the item will be found and ready by a given deadline 90% of the time. In addition, such information can be used as inputs to a return on investment analysis associated with investments that can reduce item location uncertainty. However, in order to do that, we first have to be able to calculate both parameters of the lognormal distribution probability density function (pdf), the mean μ and standard deviation σ.

Devore (2011) defines "a nonnegative random variable (rv) is said to have a lognormal distribution "if the rv $Y = \ln(X)$ has a normal distribution". The equation below gives the lognormal cumulative distribution function (cdf) of a lognormal rv when $\ln(X)$ is normally distributed with parameters μ and σ (Devore, 2011).

$$F(x;\ \mu, \sigma) = P\left(Z \leq \frac{\ln(x) - \mu}{\sigma}\right) for\ x \geq 0 \tag{12}$$

Where Z is the standard normal random variable.

Using the fact that the lognormal probability distribution is a good fit for the search times in small configurations of storerooms for both repositioning policies, one can find the probability that the search time is at a certain value. For example, for a storeroom with $k = 2$ and $h = 6$ and repositioning policy 1, the probability that search time is less than or equal 59 time units is equal to 0.8. Applying the Lognormal distribution with mean $\mu = 2.48979$ and standard deviation $\sigma = 1.53724$ for the same example of $k = 2$ and $h = 6$, the probability that search time is less than or equal 59 time units is equal to 0.849. The values of the two probabilities, probability from the actual search time distribution and the approximate lognormal distribution are very close to each other. The main advantage to approximating the search time distribution for small configurations with the lognormal probability distribution is that existing software (like EXCEL, Minitab) have built in functions. For larger storerooms, where the lognormal distribution is not a good fit, a discrete probability distribution of search times can be found by using the search heuristic to provide the search time for all of the cells.

Results exhibited in Table **3** show that there is no advantage of repositioning outside the storeroom grid to avoid put-back during the search cycle in an attempt to save time during the search process. Results proved quite the opposite; repositioning outside the storeroom not to

the nearest open aisle location inside the grid will waste more time and result in higher search time on average (higher expected search time) and will not even lower the variability between search times of different locations in a certain storeroom.

Conclusion and Future Research

In this chapter, we study a manual dense storage environment with an inverted T configuration that can be entirely defined in terms of an accessibility constant and the length of the storeroom. We propose a search plan heuristic for a single item with unknown location in an inverted T dense storage. Two repositioning policies were studied, the first repositions pallets to temporary locations in the aisles of the storeroom, which requires the need to put-back of repositioned pallets to their old locations at certain intervals during the search process to avoid the impedance of a travel path to the input/ output location upon retrieval of the item. The second policy repositions pallets outside the main storeroom grid to eliminate the need to put-back during the search process.

Test results show that the expected search time for a single item in such an environment increases with the increase in the storage density, and with the increase in the number of deeply stored items that do not have direct access to open aisle locations. Results show as well the values of the expected search time and standard deviation of search times for each storeroom configuration are close in value, with the value of the latter sometimes larger than the that of the former. The high variability in search times is due to the cumulative nature of searching, where some items that are early in the search sequence and are directly accessible have small search times, while other items require repositioning as well as efforts associated with searching other locations earlier in the search sequence. Finally, we find that a repositioning policy that uses the open aisle locations as temporary storage locations and requires put-back

of these items while searching is recommended as it results in lower expected search time and lower variability than a policy that uses available space outside the storage area and handles put-back independently of the search process.

Results from both repositioning cases show that the probability distribution of search times in a storeroom with small configurations in terms of the accessibility constant ($k = 2$ maximum) and length ($h = 9$ maximum) can be approximated into a lognormal probability distribution with a certain mean and standard deviation. Knowing the probability distribution provides the decision makers with the full range of all possible probabilities of search times, which is useful in downstream planning operations.

Future extensions can build on both the methodology and the results obtained in this research by increasing the number of requested items per order, i.e. instead of requesting and retrieving a single item, the order can consist of multiple items. Another extension is to have multiple searchers or order pickers working together to find a retrieve a single or multiple items in the same storeroom. A third extension can be to consider a system with several pallets stacked over each other in addition to being stacked on the floor, or to extend the layout beyond a single inverse T configuration. A fourth extension can be to study the search problem of a single item in a depleting system instead of a non-depleting system like the one studied here. Finally, we recommend investing in automated material handling systems that are designed specifically for a dense storage system with large number of stored items, and would be able to facilitate tailored tracking and identification systems that lowering the need of searching for a requested item. Our work can be used as an input to a Return-on-Investment model to justify such automation.

References

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

BENGÜ, G. (1995). An optimal storage assignment for automated rotating carousels. *IIE transactions, 27*(1), 105-107.

CJOSCOE. (2012). *An Introduction to Joint Operatiosn on and from the Sea*. Retrieved from http://www.cjoscoe.org/

Clark, A. V. (2002). *Sea Power 21 Series, Part I: Projecting Decisive Joint Capabilities*.

Devore, J. (2011). *Probability and Statistics for Engineering and the Sciences*: Cengage Learning.

DoD. (2014). *Startegy for Improving DoD Asset Visibility* Retrieved from

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

Frost, J., & Stone, L. D. (2001). *Review of search theory: advances and applications to search and rescue decision support*. Retrieved from http://goo.gl/nQxz4U

Futcher, F. W. (2003). *Selective Offload Capability Simulation (SOCS): An Analysis of High Density Storage Configurations*. Retrieved from http://goo.gl/vMZtVQ

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

Gue, K. R. (2006). Very high density storage systems. *IIE Transactions, 38*(1), 79-90.

Hariharan, S., & Bukkapatnam, S. T. (2009). *Misplaced item search in a warehouse using an RFID-based Partially Observable Markov Decision Process (POMDP) model.* Paper presented at the Automation Science and Engineering, 2009. CASE 2009. IEEE International Conference on.

Kierstead, D. P., & DelBalzo, D. R. (2003). A genetic algorithm applied to planning search paths in complicated environments. *Military Operations Research, 8*(2), 45-59.

Koopman, B. O. (1956). The theory of search. I. Kinematic bases. *Operations Research, 4*(3), 324-346.

Lau, H. (2007). *Optimal search in structured environments*: University of Technology, Sydney.

Lau, H., Huang, S., & Dissanayake, G. (2005). *Optimal search for multiple targets in a built environment.* Paper presented at the Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on.

Law, A. M. (2007). *Simulation Modeling and Analysis*: McGraw-Hill.

Lerher, T., Sraml, M., Potrc, I., & Tollazzi, T. (2010). Travel time models for double-deep automated storage and retrieval systems. *International Journal of Production Research, 48*(11), 3151-3172.

Lössner, U., & Wegener, I. (1982). Discrete sequential search with positive switch cost. *Mathematics of Operations Research, 7*(3), 426-440.

Matson, J. O., Sonnentag, J. J., White, J. A., & Imhoff, R. C. (2014). *An Analysis of Block Stacking with Lot Splitting.* Paper presented at the IIE Annual Conference. Proceedings.

Minitab, I. (2010). Minitab 16 Statistical Software. Retrieved from www.minitab.com

Parikh, P. J., & Meller, R. D. (2010). A note on worker blocking in narrow-aisle order picking systems when pick time is non-deterministic. *IIE Transactions, 42*(6), 392-404.

Parsons, D. (2013, March). Marine Corps Struggles With Sea-Based Supply Lines. *National Defense*.

Rana, K. (1990). Order Picking in Narrow-Aisle Warehouses. *International Journal of Physical Distribution & Logistics Management, 20*(2), 9-15. doi:doi:10.1108/09600039010005133

Scala, N. P., Jennifer. (2015). *A Value Model for Asset Tracking Technology to Support Naval Sea-based Resupply* Working Paper.

Stone, L. D. (1983). The process of search planning: current approaches and continuing problems. *Operations Research, 31*(2), 207-233.

CHAPTER FIVE: SEARCH PLAN FOR MULTIPLE ITEMS IN AN INVERTED T DENSE STORAGE SYSTEM

Abstract

In this chapter we extend the work presented in chapter 4 by studying the problem of searching for multiple items instead of just a single item in an inverted T *k*-deep *s*-stacked dense storage system with uncertainty of item locations using a single searcher. This is a much more challenging environment than the one studied in chapter 4, because to gain access to some items, not only may other items need to be moved out of the way, but also the overall number of movements for items within the system will be highly affected by the number of items stacked over each other. In addition, the searcher is given a task that includes searching and retrieving a set of items, rather than just one item. Alike to chapter 4, a search heuristic is presented that considers how to conduct repositioning, put-back and traveling within a search procedure; however, unlike chapter 4 items' retrieval is considered as well, because the procedure will not end until all *M* items that need to be found are retrieved. The repositioning policy that uses the open aisle locations as temporary storage locations and requires put-back of these items while searching is studied here, as it was the superior policy in terms of results from the previous chapter as it resulted in lower expected search time and lower variability than the policy that used available space outside the storage area and handled put-back independently of the search process. Statistical analysis of the results of the studied repositioning case is conducted as well.

Introduction

Two ways to increase the storage density are to utilize both horizontal and vertical space effectively. Increasing the lane depth by creating a deep storage system increases horizontal

space and stacking pallets on top of each other increases vertical space. In this work we study the problem of searching for *M* items in an inverted T *k*-deep *s*-stacked dense storage system with uncertainty of item locations using a single searcher.

Stacking pallets several high usually occurs while creating block stacking areas in which pallets of the same SKU are stacked on the floor in a deep lane and several pallets high (Goetschalckx & DONALD RATLDFF, 1991). However, unlike block stacking, in this research we are assuming that the pallets that need to be densely stored can be of different types of SKUs, which can require movement of pallets to gain access to other more deeply stored pallets.

Again, and as we did in the previous chapter, we are relaxing the assumption that exact locations of stored items are known to the picker, which is common in in dense storage literature. Not knowing exact items' locations forces the picker to search for the requested item before the picker can retrieve it, which increases the overall complexity of the studied problem.

Another addition in this contribution to the previous one, is assuming the need to retrieve multiple or *M* non-identical items, not just a single one before ending the search procedure. By searching for multiple items in the same trip will increase the efficiency over searching for a single item and also more accurately represent the search environment. Therefore, the picker or the searcher will be forced to search multiple items and retrieve one item after the other to the input/ output (I/O) location after finding it in the dense storeroom until he retrieves the last of item *M*. The searcher may need to move items out of his way during the search process to gain access to deeply stored stacks of pallets and remove pallets from on top of each other within the same stack to gain access to pallets stored in the lower levels of the stack.

Figure 9 is the same as the one provided in the previous chapter and it shows different elements of the search procedure for an item in a dense storage environment. The red arrow represents

the travel component during the search procedure, the blue arrow represents the repositioning effort carried out by the searcher in order to gain access to deeply stored stacks in such environments, and the orange arrow represents the put-back effort that might be carried out by the searcher at different points of time during the search sequence in order to put-back repositioned items in stacks to their old storage locations before continuing with the search procedure.

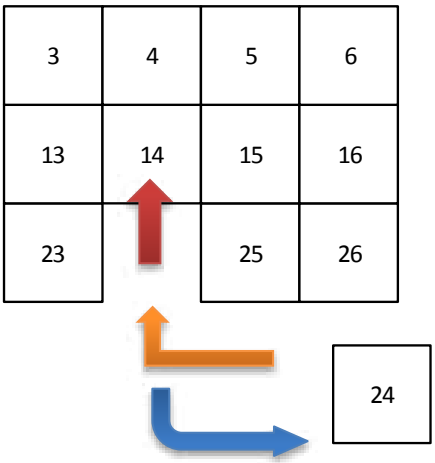

Figure 9: Components of the search procedure

Figure 9 shows a 2-D, top view of the system under study here, i.e. the figure does not provide much details about the effect of stacking high on the search components. Figure 10 exhibits an exploded partial side view of a k-deep s-stacked dense storage system with $k = 2$ and $s = 3$. All three searching components are assumed to occur while moving from one pallet within the stack to the next. Certain assumptions are made to make such inter-stack movements resemble a practical scenario, but still are consistent with the original searching definitions of search components.

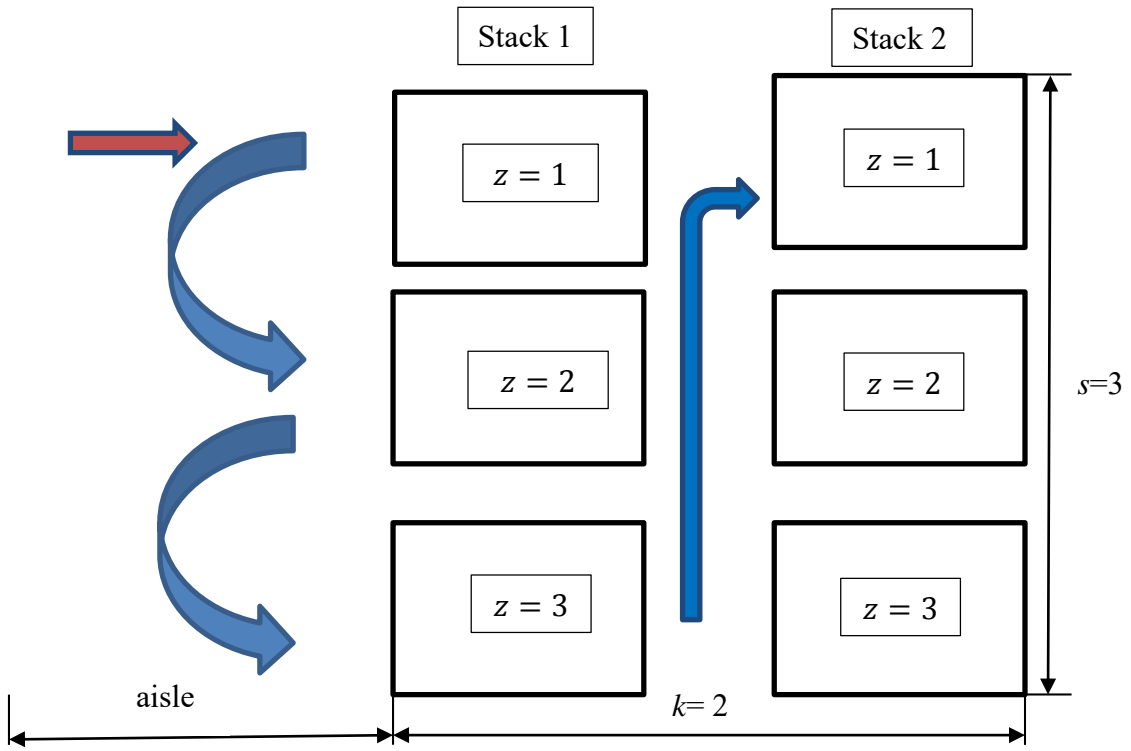

Figure 10: Side view of the k-deep s-stacked dense storage system

The exact location of a certain pallet in any given stack is defined by the x and y coordinates of the stack and the z coordinate of the pallet within the stack. The z coordinate increases in the direction towards the floor. In Figure 10, the searcher will travel to stack 1 coming from his last location through the open aisle and the first pallet that will be searched is the pallet in stack 1 with $z = 1$. Then in order to gain access to the next pallet in stack 1 with $z = 2$, the searcher will have to travel, reposition and put-back if necessary; however, as traveling only occurs in open aisle locations, the travel time will be assumed equal to zero, but there will definitely be repositioning, where pallet in stack 1 with $z = 1$ will be repositioned to nearest open aisle location that will not impeded the searcher's way to the I/O point in the future. This is denoted via the arrow connecting the two pallets with $z = 1$ and $z = 2$ in blue. The same thing will occur in order to gain access to the pallet with $z = 3$ in stack 1. The searcher will continue to the next pallet in the next deeply stored stack, stack 2 in Figure 10, if he has not

found the item he is looking for yet in the sequence of total M items. Again, repositioning effort is exerted in order to reposition the pallet at $z = 3$ in stack 1 to gain access to stack 2 and top pallet at $z = 1$ specifically.

In this contribution there are two things that are considered as novel additions to existing warehousing literature. Firstly, assuming that stacked pallets, whether those stacked in lanes on the floor or on top of each other, are not from the same SKU, i.e. all items are non-identical. Secondly, adding the search component as we did in the previous chapter to such extremely complex environment. Searching in this new warehousing environment will require moving deeply stored pallets on the floor as well as moving vertically stored pallets.

We develop analytical search plan models to estimate search times for M items in a k-deep s-stacked dense storage system, given an inverse-T configuration of the system, imperfect visibility of items' locations due to the absence of tracking and identification devices, and the use of manual order retrieval system. The purpose of this chapter is four-fold:

1. Extend our previous study of the search problem in a warehousing environment, specifically a dense storage environment by increasing the storage density through stacking pallets on top of each other. This adds complexity when studying searching in a dense storage environment, because items have to be moved from on top of each other, as well as from in front of them, in order to gain access to other deeply stored items.
2. Develop a search plan heuristic for a multiple items in a dense storage environment with an inverted T configuration. The repositioning policy that gave better results in the previous chapter is studied and a search plan heuristic is developed for it. The repositioning policy assumes items can be repositioned in open aisle locations in the

storeroom grid. In such a policy, the items need to be put-back during the search procedure to prevent blocking the way to the input/output point.

3. Derive each component of the search time equation, and use it to calculate the search time for each item in the storeroom, as well as the expected search time and standard deviation for each storeroom configuration.
4. Provide insights about the design of searching in very high density storage systems with stacking and searching for multiple items, when item location uncertainty exists. This will be done through analyzing the results and propose search plans, which can be easily articulated to human searchers, for the proposed repositioning policy.

The remainder of this chapter is organized as follows: next section surveys the literature related to the studied contribution. After that we provide the problem formulation and the assumptions of the model. A search heuristic for M items in an inverted T storeroom configurations is provided following the problem formulation. Then a summary of the numerical experiments and statistical analysis is presented. And finally, conclusions and suggestions for extending this research are provided.

Literature Review

Three bodies of literature provide a foundation to the research in this chapter. The first two bodies of literature are related to dense storage and search theory respectively and both have been surveyed in previous chapters. The third body of literature is the concept of block stacking and it is surveyed here.

Block stacking is adopted when high utilization of storage space is needed and large quantities of boxes or pallets are to be stored (Goetschalckx & DONALD RATLDFF, 1991). Block stacking is usually used when large numbers of one or few SKUs are need to be stored, and

this is done through placing units in rows on the floor and on top of one another. Stacks are placed several lanes deep perpendicular to the access aisle (Matson, Sonnentag, White, & Imhoff, 2014).

Research on block stacking focused mainly on answering questions about the depth of a storage row. According to Matson et al. (2014), two ways to manage storage rows can be found in literature. The first way is dedicated storage in which each row is assigned for a single SKU. The second way is random or shared storage in which rows are made available to any SKU as soon as they are empty. Random or shared storage has been the main focus of research up to date including the work of Matson et al. (2014).

One of the early works to determine the lane depth for a single product or a single lane depth in a block stacking environment is the work of Kind (1965, 1975). In this work, a formula to determine a single lane depth is proposed, which was found to "overestimate the lane depth" by Matson and White (1981).

Kooy (1981) introduced the term "perfectly balanced" warehousing system. In this work, the optimal number of lanes of each depth for a certain warehouse design is determined. However, White, Sonnentag, and Matson (2013) studied the effect of adding transportation cost to the storage cost in the problem of determining the optimal storage depth, and they found that this addition makes the "row shallower".

BERRY (1968) analyzed the throughput of a block stacking design in addition to space utilization. They concluded that the optimal row depth for "fast-moving" SKUs is greater than that for "slow-moving" SKUs.

A much more recent study on block stacking is the work of Matson et al. (2014), in which the authors studied both the dedicated and random storage environments, with the objective of minimizing a certain total cost function using a dynamic programming solution approach.

In this research we do not focus on studying block stacking from a mathematical modeling point of view. However, we are much more interested in the idea of items' storage in a stackable manner. Unlike, most of block stacking literature, we are assuming having number of SKUs equal to the total number of items, i.e. each cell in our storeroom holds a unique item or a different SKU. Finally, the lane depth in our problem is fixed which is assumed to be equal to the accessibility constant and the number of aisles is fixed as well, with one vertical aisle and one horizontal aisle interesting to form and inverted T.

Problem Formulation

The problem under study is a search planning problem with the objective of minimizing the expected time required to find and retrieve M different items given imprecise knowledge of their exact storage locations in a k-deep, s-stacked dense storeroom with an inverted T configuration. The time to find or to detect the location of each item of the M items might involve the time required to move other items out or from on top of each other during the search process. Thus, the search time is the combination of travel, repositioning, and put-back times as previously defined. Here we are interested in minimizing the expected search time for each storeroom configuration defined by k, h, s and M.

In addition to the search time, there is the retrieval time which is the amount of time that will be spent retrieving each of the M requested items to the input/ output (I/O) location. Retrieval time of a certain item of the M items will be affected by the location of each of that item in the storeroom and the size of the storeroom in terms of k and h. Vertical travel time is neglected in

our analysis. We are interested in calculating the average or expected retrieval time for one or more items from the storeroom under examination.

The sequence in which all occupied cells or stored items in the storeroom will be visited is called the search sequence array Ψ. The M items that need to be retrieved are listed in the sequence in which they are visited in the sequence array Ψ. Therefore, the number M items to be retrieved is less than or equal to the number of cells listed in the array Ψ.

The following assumptions are made about the studied problem:

- When the search process starts, there is an equal probability of having the item stored in any location or cell.
- The number of searchers in the studied system is equal to one.
- The order to retrieve M items is received at the input/ output (I/O) location, which is assumed to be fixed and with known coordinates.
- The searcher will retrieve an item among the M requested as soon as he finds it and he will continue his search for the remaining requested items from the last location he stopped at during the search procedure. This process will continue until all M items are found and retrieved.
- Items are stacked deep on the floor, such that not all items are directly accessible from an aisle. Items are also stacked high above the floor in the vertical direction on top of each other with no supporting rack structure separating one item from the ones stacked above it. Each item occupies one cell. Each cell represents a single non-overlapping search region of equal size. All cells' locations are known with certain x, y and z coordinates.

- The studied system is assumed to be a non-depleting storage system, i.e. cases are picked off pallets. The number of items present in the system remains the same. Therefore, the inverted T configuration of the storeroom remains the same after items' retrievals.

The studied environment is different from the one studied in chapter 4, thus, detailed assumptions about travel, repositioning, put-back and retrieval times will be presented in upcoming sections.

The coordinate system used before is adopted here, where the system is defined entirely by the three variables k, h and s. In the horizontal direction the origin point ($x = 0$ and $y = 0$) is assumed to be located at the intersection between the vertical and horizontal aisles, and the I/O cell is located at $x_j = -k$ and $y_j = 0$. Figure 11 shows a 2-D view of sample system with $k = 2$ and $h = 6$.

3	1	2	3	4	5
2	6	7	8	9	10
1	11	12	13	14	15
0	-k = -2 (16)	-1 (17)	0 (18)	1 (19)	k = 2 (20)
-(k-1) = -1	21	22	23	24	25
-k = -2	26	27	28	29	30

Figure 11: Sample Inverted T system with $\boldsymbol{k = 2}$ and $\boldsymbol{h = 6}$

The definition of the coordinated system explained above has to be extended in order to accommodate the items stacked in the vertical direction as follows:

- The height of the stack is defined by the value *s* or the number of stacked items and all stacks in the storeroom are assumed to have the same height.
- Items which belong to the same stack have the same *x* and *y* coordinates and different z coordinate. The z coordinate starts at $z = 1$ at the top most pallet within the stack and increases in the floor direction to $z = s$ for the lower most pallet in the stack.
- The storage density was defined previously as the ratio of the number of stored pallets *n* to the total number of cells *N*, i.e. $n/_N$. This is a 2-D storage density as it ignores vertical space utilization. Therefore, we define a 3-D storage density, which takes into consideration the utilization of both horizontal and vertical space and will depend on the height of the stack *s*. The 3-D storage density is given by $\frac{n'}{N'} = \frac{n.\varsigma}{N.s}$. Where, n' is the total number of pallets in the system, N' is the total number of potential storage locations in both horizontal and vertical directions (which includes aisle locations and the space above them), *n* is the number of pallet stacks, ς is the number of items stacked over each other, *N* the total number of potential storage locations in the horizontal direction (which includes aisle locations), and *s* is the maximum height of the stack for each storeroom configuration.

For example, for the system shown in Figure 11, $n = 22$ and $N = 30$. Assume that the maximum allowed stack height $s = 3$ and the actual stack height $\varsigma = 1$. Therefore, the total number of pallets in the system $n' = 22 \times 1 = 22$, $N' = 30 \times 3 = 90$, and the 3-D storage density $= \frac{22\times1}{30\times3} = 0.244$. This storage density is less than the 2-D storage density when the maximum allowed stack height *s* is equal to the actual stack height ς and both equal to 1, i.e. $\frac{n}{N} = \frac{22}{30} = 0.733$, or when ς takes the same value of *s*.

In this work ς will always be assumed equal to s and the storage density can be simply calculated by $\frac{n\prime}{N\prime}$, or in terms of k and h: $\frac{h(2k+1)-(2k+1)-(h-1-k)}{h(2k+1)}$.

The set of cells C is still divided into the set of occupied cells C_O and the set of unoccupied cells C_U. For the set C_O each cell is occupied by a stack of pallets of height s not just a single pallet as in chapter 4 and the set C_U represents cells forming the open aisles. As in the previous chapter, the set of occupied cells includes five disjoint subsets; $C_O = C_O^{UL} \cup C_O^{UR} \cup C_O^{BL} \cup C_O^{BR} \cup C_O^{BC}$, where, C_O^{UL} is the set of occupied cells with stacks in the upper left part of the storeroom including all items within those stacks. Similarly, C_O^{UR} for the upper right part of the storeroom, C_O^{BL} for the bottom left part of the storeroom, C_O^{BR} for the bottom right part of the storeroom, and C_O^{BC} for the bottom center part of the storeroom.

The search plan problem here is modeled as a network as we did in the previous chapter. The nodes of the network represent the cells being searched in the vertical or horizontal directions, and the weights of the arcs represent the summation of the time needed to travel between a pair of cells, plus the time needed to reposition pallets blocking the searcher's access to a certain cell (if any), plus the put-back time of already repositioned pallets to their storage location before traveling to cell *j* coming from cell *i* (if any). The weights of arcs are calculated whether the cells connected by these arcs belong to the same stack or not. The search time equation and its components is given by Equation 1 in chapter 4. However, there are some differences between the studied environment here and the environment examined in chapter 4, therefore, the travel, repositioning and put-back components of the search time equation will be discussed in much more details in the following sections.

Travel Time

Travel time is the time taken by the searcher to travel through open aisle locations from cell i to cell j. Travel time between cell i and cell j is denoted by $\tau_{(x_u,y_u),(x_v,y_v)}$, where cell u with coordinates (x_u, y_u) is the nearest open aisle location to cell i, and cell u with coordinates (x_v, y_v) is the nearest open aisle location to cell j.

We derive the coordinates of the pair of nearest open aisle cells u and v for each pair of cells i and j in the same manner explained in chapter 4.

The only addition we need to introduce to the definition of travel time here is the travel time between items which belong to the same stack. Any pair of items belong to the same stack of items share the same x and y coordinates, i.e. $x_i = x_j$ and $y_i = y_j$, therefore, $x_u = x_v$ and $y_u = y_v$, consequently, the travel time between cells i and j is equal to zero, i.e. $= \tau_{(x_u,y_u),(x_v,y_v)} = 0 \; \forall \; z \; coordinates \; having \; x_i = x_j$ and $y_i = y_j$.

Repositioning Time

Repositioning time (R_j) is the time needed by the searcher to move a pallet out of the way in order to gain access to pallet j. One repositioning policy will be investigated here, where pallets are temporarily repositioned to inside the storeroom to an unoccupied cell in a manner that will not impeded the searcher's future path to the I/O location.

Assumptions about the repositioning policy made here are similar to the ones made in chapter 4. However, the major difference is having s pallets stacked over each other at each location j. Therefore, two distinct repositioning moves have to be defines here. The first repositioning move is repositioning of all items belonging to a certain stack in order to gain access to another

deeply stored stack of pallets. For example, in Figure 10, in order to gain access to the deeply stored stack of items number 2, all pallets in stack 1 have to be repositioned out of the searcher's way one pallet at a time, or *s* number of times, where *s* is the stack height.

The second repositioning move, is repositioning in order to gain access to stored pallets within the same stack of pallets (i.e., repositioning pallets stacked on top of a pallet). For example, in Figure 10, the searcher will not gain access to pallet in stack 1 with $z = 2$ unless pallet in $z = 1$ on top of it is repositioned. The number of repositioning moves in order to gain access to the lowest pallet in a vertical stack is equal to $s - 1$. In Figure 10, in order to gain access to the lowest stored pallet at $z = 3$ in the vertical stack 1, the two pallets on top of it have to be repositioned. Therefore, the number of repositioning moves is equal to $s - 1 = 3 - 1 = 2$.

There are a few things that characterize repositioning in the studied system:

- Pallets are repositioned inside the storeroom to open locations that will not impeded the searcher's future way to the I/O location and they are put into stacks of the same height *s* as the stacks they originally belong to.
- When repositioned into new stacks, the pallets' vertical order is inverted, as the top most pallet is the one to be repositioned first, which ends up at the bottom of the new repositioned stack.

<u>Subroutine Reposition</u>

Repositioning to gain access to deeply stored stack

If (cell $j^* \in$ to stack in C_o^{UL}) AND $(x_{j^*} \leq -2)$

Then start repositioning, and set $Repo[w] = 1$

Else No repositioning needed, set $Repo[w] = 0$

If (cell $j^* \in$ to stack in C_o^{UR}) AND $(x_{j^*} \geq 2)$

Then start repositioning, and update $Repo[w] = 1$

Else No repositioning needed, set $Repo[w] = 0$

If (cell $j^* \in$ to stack in C_o^B) AND $(y_{j^*} \leq -2)$

Then start repositioning, and set $Repo[w] = 1\ 1$

Else No repositioning needed, set $Repo[w] = 0$

Repositioning to gain access to a pallet within a stack

If $z_{j^*} > 1$

Then start repositioning, and set $Repo[w] = 1$

Else No repositioning needed, set $Repo[w] = 0$

Put-back Time

Put-back time (P_{bj}) is defined as the time needed to move repositioned pallets back to their storage locations before searching cell j. Put-back of repositioned pallets has to be done at certain points of time during the search procedure in order to prevent accumulation of pallets in open aisle locations to the point that the searcher's path to the I/O location will be impeded upon retrieval of the requested pallet.

The proposed put-back rule is characterized by the following:

- Repositioned pallets are put-back to the storage locations before moving to the next cell if the number of repositioned pallets is equal to the number of maximum allowed pallets to be repositioned, which is equal to $k \times s$, where k is the accessibility constant and s is the stack height or the maximum allowed number of pallets to be stacked over each other.
- Put-back of pallets in a temporary repositioning stack is done one by one, i.e. put-back time is multiplied by s.

For an already repositioned pallet j to be put-back to a storage location belonging to the sets $C_O^{BR}, C_O^{BC}, C_O^{BL}$ or C_O^{UR} with coordinates x_j and y_j, the put-back time is calculated using Equation 13.

$$P_{b_j} = 2 \times s \times \left[|y_j| + |x_j + k + 1| - 1\right] \tag{13}$$

For an already repositioned pallet j to be put-back to a storage location belonging to the set C_O^{UL} with coordinates x_j and y_j, the put-back time is calculated using Equation 14.

$$P_{b_j} = 2 \times s \times \left[y_j + |x_j| + |-k - 1| - 1\right] \tag{14}$$

Subroutine put-back defined in chapter 4 will exhibit a slight change as follows:

<u>Subroutine Put-back</u>

If $\sum_{l=\emptyset}^{w} Repo[l] = \alpha sk$

Then start put-back, and set $\alpha = \alpha + 1$
Calculate put-back time using Equations

Else no Put-Back required

This subroutine counts the number of repositioned pallets that have not been put-back using the array $Repo$. Where, array Repo is the array takes the value 1 to indicate a repositioned pallet and 0 to indicate a pallet that does not need repositioning, α is used to count the number of put-back procedures, and sk is the maximum allowed number of repositioned pallets after which repositioned pallets have to be put-back before continuing with the search procedure, otherwise the repositioned pallets will block the path to the I/O point.

It is worth mentioning that unlike the repositioning procedure, the put-back procedure puts all items in a stack of repositioned pallets back to their storage location one by one but will not stop until sk pallets have been returned to storage locations. On the other hand, repositioning

is initiated only whenever the searcher needs to gain access to a deeply or lower stored pallet, whether within the same stack or a complete deeply stored stack of pallets.

Retrieval Time

Deriving retrieval time equations is important in this contribution because we are considering searching and retrieving M pallets from the storeroom, not just a single pallet as we did in chapter 4. The importance of considering retrieval while the search procedure is being carried out lies in the fact that there will be $M-1$ retrievals of $M-1$ pallets from their storage locations to the I/O location before finding the last pallet or pallet M and retrieving it.

All M pallets will be retrieved and brought to the I/O location that has a fixed predefined coordinates $x=-k$ and $y=0$. Retrieval distances will depend on the x and y location of pallets being retrieved and the pallet's location within the stack or the z location is ignored. For any pallet belongs to stack located in cell j $\{x_j, y_j\}$ belonging to the sets $C_O^{BR}, C_O^{BC}, C_O^{BL}$ and C_O^{UR}, the retrieval distance is given by:

$$\rho_{(x_j,y_j)} = 2 \times \left[\left|y_j\right| + \left|x_j + k\right|\right] \tag{15}$$

For any pallet belongs to stack located in cell j $\{x_j, y_j\}$ belonging to the set C_O^{UL}, the retrieval distance is given by

$$\rho_{(x_j,y_j)} = 2 \times \left[y_j + \left|x_j\right| + k\right] \tag{16}$$

Retrieval times above are multiplied by 2 because during the retrieval of pallets from the storeroom, the searcher is supposed to return back to the last visited location to continue with the search procedure.

Search Plan Heuristic

The search plan heuristic starts by generating arcs from the current cell (initially, when a new order of *M* items is received, the current cell is the I/O cell) and evaluating the weights (function of travel, repositioning, and put-back times) of each of those arcs. The cell connected with the arc with the least weight is selected. The indices of the cell connected with the arc with the least weight are added to the array Ψ of visited cells. This procedure is repeated until the first of *M* requested items is found, then this item is retrieved and taken to the I/O location. The search procedure is then continued from the last visited location, and the search continues for the remaining $M - 1$ items until all of them are found and retrieved.

All pallets in the storeroom are considered as unique items, therefore, all *M* items on the request list received by the searcher will be different. The unordered array Ω of requested pallets of size *M* is defined and contains all pallets that are requested. The search procedure will stop only if all of the pallets in the array Ω have been found and retrieved. The array ℝ is defined as the array of found items during the search procedure. ℝ is initially assumed to be empty and the search procedure does not stop until $|\mathbb{R}| = M$.

The search plan heuristic can summarized through the following pseudo code:

1. Given: k, h, s, M, Ω and ℝ.
2. Generate x,y and z coordinates for each pallet in the storeroom, classify each pallet $i \in C$ in its appropriate subset ($C_O^{UL}, C_O^{UR}, C_O^{BL}, C_O^{BR}, C_O^{BC}, C_U$).
3. Determine for each pallet occupying cell $i \in C_O$ the open aisle location u.
    a. If ($i \in C_o^U$ and $|y_i| \neq 1$) Then $x_u = 0$ and $y_u = y_i$
    b. If ($i \in C_o^U$ and $|y_i| = 1$) Then $x_u = x_i$ and $y_u = 0$
    c. If ($i \in C_o^B$) Then $x_u = x_i$ and $y_u = 0$
4. Start with an empty visit sequence array Ψ, empty set of visited cells array ψ, and empty set of found items array ℝ . Set $w = 0$; $\alpha = 1$, and FOUNDCOUNTER=0
5. Start the search procedure at I/O cell, i.e. $\Psi[w] = I/O$ cell location
6. Set $i = I/O\ cell$.
7. Calculate the distance $T_{i,j} = \tau_{u,v} + R_j + P_{b_{i,j}}$

Between cell i and cell $j \; \forall \; j \notin \psi$ using travel, repositioning and put-back equations.

8. Select cell $j^*$with min T_{ij}; break ties arbitrarily
9. Update $\Psi[w] = j^*$ and $\psi = \psi + j^*$
10. Travel from the nearest open aisle location of cell i to the nearest open aisle location of cell j^*
11. Do Subroutine *Reposition*
12. Do Subroutine *Put-back*
13. If ($j^* \in \Omega = \Psi[w]$)

    Then retrieve item $\Psi[w]$ to the I/O location.
    Set FOUNDCOUNTER = FOUNDCOUNTER +1.
    Update $\mathbb{R} = j^*$.
    If ($FOUNDCOUNTER = |\Omega|$)
    Then stop *search*
    Output: $\Psi[w]$.

Else

$w = w + 1, i = j^*$and go to step 7

<u>Expected Search and Retrieval Times</u>

The search plan heuristic above gives us the search sequence array Ψ[w], or in other words the sequence of visited cells until we find all the M pallets given in Ω, and the search does not stop until all M requested pallets are found and retrieved. We are interested in minimizing the summation of the expected search time and the expected retrieval time for each storeroom configuration defined with k, h, s and M.

For each storeroom configuration defined with k, h, s and M, the M requested pallets can be located at any of the n' occupied vertical or horizontal locations, and each of the M requested pallets have an equal probability to be located anywhere within the storeroom. Therefore, in order to calculate the expected search time a probability distribution has to be defined.

For a search sequence ordered array $\Psi[w]$ with $w = [0, 1, 2, \dots, n']$. The cumulative probability or the cumulative distribution function (cdf) of finding M items given the search sequence

requires visiting i or less cells (where i is the sequence index in the array $\Psi[w]$ and $i = 0$ is the index of the I/O location) is calculated by Equation 17:

$$F(i) = \begin{cases} \frac{\binom{i+1}{M}}{\binom{n'}{M}} & for\ i \geq M\ and\ M \leq n' \\ 0 & otherwise \end{cases} \tag{17}$$

Where $F(i)$ is the cumulative distribution function, i is the sequence index in Ψ, M is the number of pallets requested, and n' is the total number of pallets in the system.

Accordingly Equation 18 can be used to calculate the probability mass function (pmf) or the probability of finding M pallets after searching exactly i cells, which is equivalent to finding it in the ith sequence in the search sequence $\Psi[w]$:

$$p(i) = p_i = \begin{cases} \frac{\binom{i+1}{M}}{\binom{n'}{M}} - \sum_{j=0}^{j=i} \frac{\binom{j}{M}}{\binom{n'}{M}} & for\ i \geq M\ and\ M \leq n' \\ 0 & otherwise \end{cases} \tag{18}$$

For a search sequence array $\Psi[w]$ with $w = [0, 1, 2, \dots, n']$. The search time for any cell $i \in w$, is calculated by:

$$T_i = T_0 + \sum_{j=1}^{i} T_{j-1,j}\ \forall i = 0, 1, \dots, n \tag{19}$$

Where, T_i is the search time for cell i in the search sequence $\Psi[w]$, T_0 is the search time for the first cell in the sequence $\Psi[w]$, $T_{j-1,j}$ is the search time between cell with index $j - 1$ and cell with index j in the array $\Psi[w]$.

Therefore, the expected search time for M pallets, denoted as E_M is calculated by:

$$E_M = \sum_{i=0}^{n'} T_i \times p_i \tag{20}$$

The standard deviation of search times for M pallets, denoted as SD_M, is calculated by:

$$SD_M = \sqrt{\sum_{i=0}^{n'} T_i^{\,2} \times p_i - E^2} \tag{21}$$

The calculation of the expected retrieval time is a little bit different. There is an equal probability of finding any of the M pallets anywhere in the storeroom. Therefore, the expected retrieval time for a single pallet from the storeroom can be calculated by:

$$E_\rho = \frac{\sum_{i=1}^{n'} \rho_i}{n'} \tag{22}$$

Where E_ρ is the expected retrieval time for a single pallet, and ρ_i is the retrieval time for pallet i in the search sequence $\Psi[w]$.

For M pallets, the expected retrieval time for M pallets will be equal to:

$$E_{\rho_M} = \frac{\sum_{i=0}^{n'} \rho_i}{n'} \times M \tag{23}$$

Finally, for M pallets, the standard deviation of retrieval times is calculated by:

$$SD_{\rho_M} = \sqrt{\frac{\sum_{i=0}^{n'} \left(\rho_i \times M - E_{\rho_M}\right)^2}{n' - 1}} \tag{24}$$

<u>Search Plan</u>

Applying the search heuristic given in the previous section will generate the search array Ψ, which provides the sequence of cells to be visited. The assumption is when the searcher finds the first of M requested pallets, he will retrieve it and continue searching from last visited cell to find another item and so on until the last of M pallets is found and retrieved.

For the system shown in Figure 11, the search procedure starts after the searcher receives an order of size M at the I/O cell located at cell 16 $\{-2, 0\}$ with $k = 2$ and $h = 6$, and assuming that $s = 2$, the searcher then follows the search sequence given by the search array:

$$\Psi = \{16, 11, 21, 12, 22, 23, 14, 24, 15, 25, 7, 9, 2, 4, 7, 12, 23, 24, 14, 15, 25, 22, 21, 11, 28, 4, 9, 28, 29, 2, 27, 6, 26, 30, 29, 10, 27, 1, 5, 10, 30, 6, 1, 26, 5\}$$

Numbers in green in Ψ represent cells with $z = 1$, i.e. top pallets and at the same time have direct access to open aisles. Numbers in red are cell with $z = 2$, i.e. the lower stored pallets in a system with $s = 2$. Finally, numbers in black are for deeply stored pallets with $z = 1$. In this system the total number of pallets or visited locations is equal to $n' = 44$ and the total number of locations including open aisle cells is equal to $N' = 60$.

After knowing the search sequence, we can calculate the search time for each cell in a certain configuration and the expected search time for the entire system for each value of M and the corresponding standard deviation. In addition, the expected retrieval time and the corresponding standard deviation are calculated as well. Expected search and retrieval times and their corresponding standard deviations are calculated using the formulas in the previous section.

In the following section, numerical experiments are conducted in order to understand the performance of each storeroom configuration in terms of the expected search and retrieval times and their corresponding standard deviations. Knowing how each configuration will perform, will make it easy for the decision maker to decide on using a certain system with specific k, h, s and M values.

Numerical Experiments

In this section we conduct numerical experiments to study the proposed search heuristic with all its components outlined in the previous sections using a statistical Design of Experiments (DOE) approach. A set of problems is generated, which is solved using the proposed search plan heuristic. The responses of interest in our analysis are the expected search time and the standard deviation of search times.

Each problem instance represents a different storeroom configuration and is defined entirely by the accessibility constant k, the storeroom length h and the number of pallets stacked over each other or the vertical height s.

The algorithm is programmed in Python 3.4, and all experiments are conducted on a desktop computer supported by an Intel Core 2 Duo processor of 3.0 GHz CPU speed.

For the Design of Experiments, three key factors are identified, and two levels are set for each factor, thus a 2^3 factorial design with 8 different experiments is used. The three factors are:

1. **H/W**: shape factor.

The shape factor is the ratio between the storeroom length h and the storeroom width W. For an inverted T configuration, the storeroom width is equal to $2k + 1$. Therefore, the shape factor is given by $\frac{h}{2k+1}$. However, the minimum possible value of the storeroom length h is $2k + 2$ in order to maintain the inverted T configuration. Two levels of the shape factor will depend on the value of k. For low level shape factor, k is generated randomly from the discrete uniform distribution U(2, 5) and h is set to its minimum possible value $2k + 2$. And for the high level shape factor, k is generated randomly from the uniform distribution U(6, 10) and h is set to 50.

2. **S**: stack height.

High level of this factor indicates relatively large stack height and was generated from the discrete uniform distribution U (3, 4). The low level of this factor was generated from the uniform distribution U (1, 2).

3. **M**: number of pallets per order.

High level of this factor indicates large number of requested pallets and was generated randomly from the discrete uniform distribution U (6, 10). The low level of this factor was generated randomly from the uniform distribution U (1, 5).

Values for each one of the two levels of the three factors are summarized in Table 6.

Table 6: Values for each factor level

No.	**Factor**	**Low level (L)**	**High level (H)**
1	**H/W**: shape factor	k from U(2, 5) & $h = 2k + 2$	k from U(6,10) & $h = 50$
2	**S**: stack height	U(1, 2)	U(3,4)
3	**M**: number of pallets requested	U(1,5)	U(6,10)

A total of 80 experimental instances are randomly generated and each represents a different storeroom configuration. Instances were generated such that eight (2^3) different runs each with of 10 replications are conducted. The expected search times and the corresponding standard deviations of each of the 80 instances are recorded. For each of the 10 replications per experiment, the average expected search time, minimum expected search time, and maximum

expected search time are recorded. The same thing is done for the standard deviations of search time.

Results obtained from implementing the search plan heuristic on the 80 problems in terms of the expected search time and standard deviation are summarized in Table 7. Minitab (2010) statistical package was used for calculations and the analysis of results.

Table 7: Response for each of 10 replications

				Expected Search Time			**Standard Deviation**		
#	**H/W**	**S**	**M**	**Average**	**Max**	**Min**	**Average**	**Max**	**Min**
1	L	L	L	765.69	2323.30	59.46	374.79	1064.89	39.36
2	H	L	L	45699.09	146132.8	8622.4	23297.27	54475.79	5727.71
3	L	H	L	4733.31	10833.88	672.36	2837.28	8519.57	676.09
4	H	H	L	352861.36	638416.17	88073.25	177538.1	265134.1	85000.47
5	L	L	H	576.46	1137.63	90.19	132.75	243.82	30.64
6	H	L	H	60650.86	168837.45	18204.39	13428.01	35659.43	4170.29
7	L	H	H	8013.26	22852.05	935.43	1795.94	4666.68	204.11
8	H	H	H	476974.04	658756.61	280467.3	104643.8	145185.6	55106.17

The main and interaction effects, as well as the *p-values*, of the three factors on the two responses of interest, the expected search time and standard deviation, from the Analysis of Variance (ANOVA) are exhibited in Table 8 and Table 9 respectively.

Table 8: Main and interaction effects and corresponding p-vales for expected search time

No.	Factors	Effect	*p-value*
1	H/W	230524	0.000
2	S	183722	0.000
3	M	35539	0.055
4	H/W*S	178020	0.000
5	H/W*M	33993	0.066
6	S*M	28158	0.126
7	H/W*S*M	26423	0.151

Table 9: Main and interaction effects and corresponding p-vales for standard deviation

No.	Factors	Effect	*p-value*
1	H/W	78441.6	0.000
2	S	62395.6	0.000
3	M	-21011.7	0.001
4	H/W*S	60332.7	0.000
5	H/W*M	-20370.1	0.001
6	S*M	-15956.1	0.008
7	H/W*S*M	-15556.4	0.010

A positive effect value in Table 8 and Table 9 means that the response value increases with increasing the corresponding factor value, and vice versa, whether this response is the expected

search time as it is the case in Table 8 or the standard deviation of the search times as in Table 9. The relative weight of an effect is directly related to its magnitude.

From Table 8 and at 95% level of confidence, the ANOVA results show that only the shape factor, the stack height and the interaction between these two factors statistically affect the expected search time, as their corresponding *p-values* are the only ones less than 0.05 . The values of the effects of these two factors and their interaction are positive, which indicates an increase in the value of the expected search time with the increase of the value of the shape factor, the stack height or both of them simultaneously. The magnitude of the shape factor effect is the highest of the three, which means that the shape factor, and accordingly the size of the storeroom and the number of stored pallets, has the highest significant effect on the expected search time.

From Table 9, all *p-values* of all three factors and their interactions are less than 0.05. Therefore, at the same level of confidence of 95%, all three factors and their interactions are significant in the value of the response or the standard deviation of search times. The values of the effects of the three factors and their interactions are not all positive. Therefore, some of the factors are significant but have a negative effect on the value of the response or the standard deviation of search times. The main effect of the factor M and the effects of its interactions with the other two factors are all negative. Accordingly, as the number of requested pallets M per search procedure increases the variation in search times, represented by the standard deviation of search times, decreases. The two other factors, shape factor and stack height, and their interaction, have a significant but positive effect on the value of standard deviation, i.e. the standard deviation will increase of the size of the storeroom, the stack height or both of them, together.

Effect plots are also a useful tool that graphically demonstrates either the main effect of a factor or the interaction effect between two factors. The interaction effect is the average difference between the effect of the first factor at the high level of the other factor, and the effect of first factor at the low level of other factor (Montgomery, 2008). Figure 12 and Figure 13 are the main and interaction effect plots for the expected search time respectively.

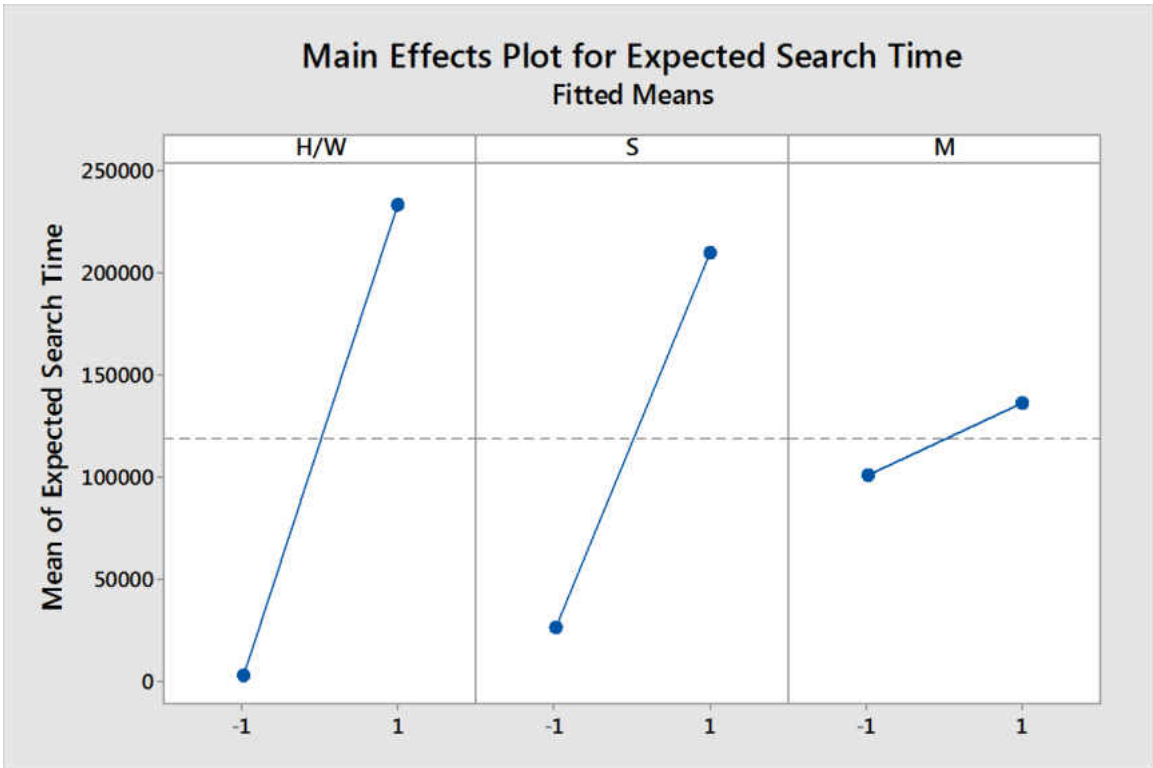

Figure 12: Main effects plots for expected search time

The horizontal dotted line in Figure 12 represents the average of the expected search time for all 80 experiments. Each of the three panels in Figure 12 represents one of the three factors, and they show that the expected search time is expected to increase with the increase in the values of any of three factors. However, the increase in the shape and stack height factors are expected to have a higher effect on the value of the expected search time compared to the

number of requested pallets. In fact, the factor M or the number of requested pallets was found insignificant at level of confidence of 95%.

Figure 13 shows that for each pair of factors, each factor does not affect the other. In other words, the performance of the search heuristic in terms of the resulting expected search time for each storeroom configuration, does not change when a certain factor changes from the low to high level while the other factor remains constant.

Figure 13: Interaction effect plots for expected search time

The main and interaction effect plots of the second response of interest, i.e. standard deviation of search times, are provided in Figure 14 and Figure 15 respectively.

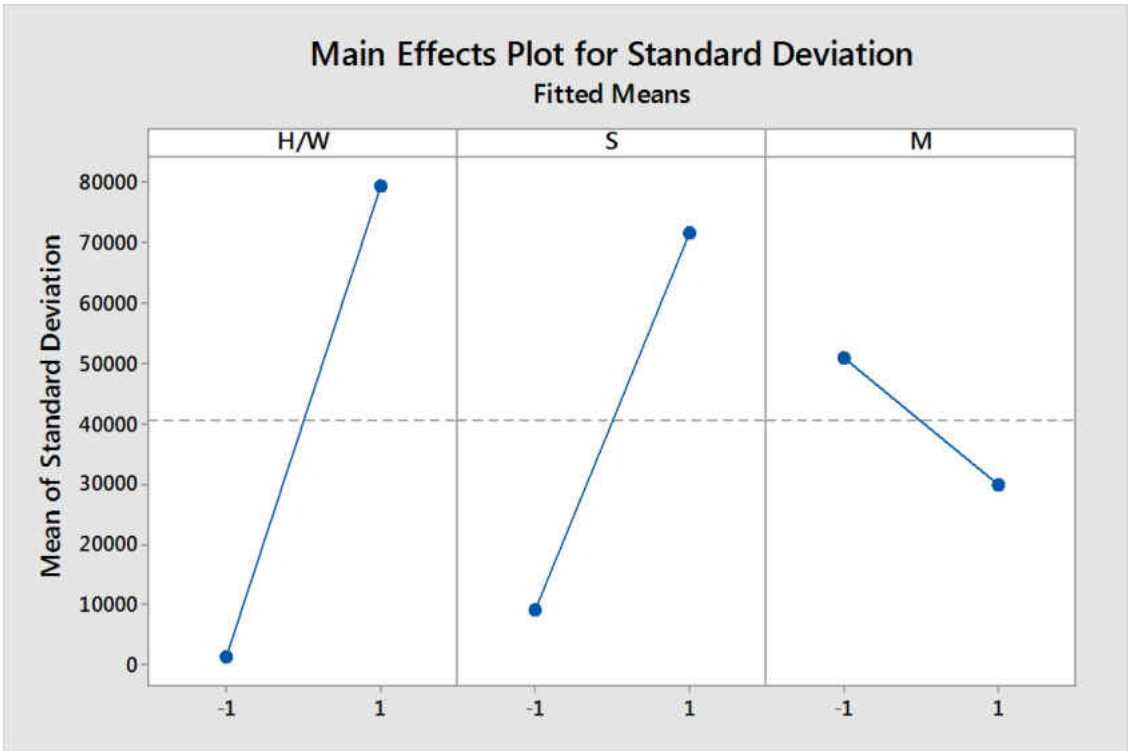

Figure 14: Main effects plots for standard deviation

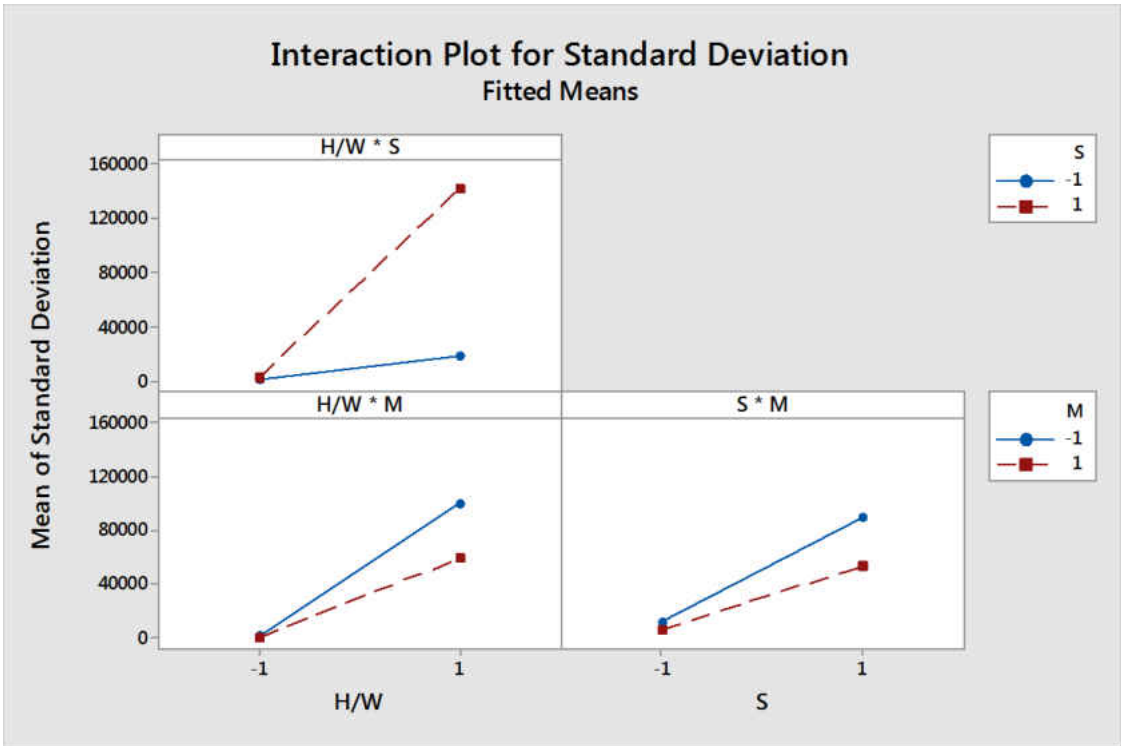

Figure 15: Interaction effect plots for standard deviation

Figure 14 shows the positive effect that both the shape and stack height factors have on the value of the standard deviation, and the negative effect the number of requested pallets M has on the value of the standard deviation. All three factors have been shown to be significant from the results shown in Table 9.

Each pair of factors does not affect the other as shown in Figure 15. Therefore, the performance of the search heuristic in terms of the resulting standard deviation of search times for each storeroom configuration, does not change when a certain factor changes from the low to high level while the other factor remains constant.

Conclusion and Future Research

In this chapter, we propose a search plan for multiple or M pallets in a manually operated dense storage environment. The storage environment is defined entirely in terms of the accessibility constant k, the storeroom length h, and the number of pallets stacked over each other vertically s.

In such a complex environment, the searcher has to move pallets out of the way to gain access to deeply stored stacks of pallets, and move pallets from on top of each other to gain access to lower pallets within the same stack. Such pallets are repositioned temporarily inside the storeroom to other locations that will not impeded the searcher's path to the Input/ Output location upon retrieval. Such repositioning policy is adopted in this chapter as it proved its superiority in terms of the expected search time among the repositioning policies studied in chapter 4. A search plan heuristic is developed that is when followed lowers the overall effort of finding and retrieving a request of M pallets from the storeroom.

A Design of Experiments (DOE) approach is adopted and Analysis of Variance (ANOVA) is used to analyze the experimental results. Results show that the performance of the proposed search plan heuristic in terms of the expected search time for any storeroom configuration is highly affected by the size of the storeroom and the number of pallets stacked in the vertical direction. A storeroom with a large size and large stacks' heights requires more locations to be searched, more pallets to be repositioned and put-back, and accordingly more search effort. On the other hand, the performance of the search heuristic is not greatly affected by the number of pallets requested.

The cumulative nature of the searching procedure results in high variability in search times as discussed in chapter 4. One way to overcome such effect is to request multiple items instead of just one item as we did in chapter 4. Experimental results show the negative effect of increasing the number of requested pallets on the variability of search times. In other words, the variations in search time for each storeroom configuration can be decreased by requesting multiple items instead of just a single item.

Knowing the effect of increasing the number of requested pallets on decreasing the variability in search times would it make it easier for decision makers to decide on a certain size of dense storerooms and the number of item to be retrieved from such a storeroom per search trip.

Future research can benefit from both the methodology and the results obtained in this chapter by investigating the possible effect of having more than one searchers working together to retrieve multiple items from the same storeroom. A second extension can be adopting the same search procedure to other storage environments not necessarily a dense storage environment with an inverted T configuration. Finally, investigating the effect of having some level of

knowledge about items' locations not complete uncertainty about locations as we assume here, i.e. different probability distribution.

References

BERRY, J. R. (1968). Elements of warehouse layout. *The International Journal of Production Research, 7*(2), 105-121.

Goetschalckx, M., & DONALD RATLDFF, H. (1991). Optimal lane depths for single and multiple products in block stacking storage systems. *IIE TRANSACTIONS, 23*(3), 245-258.

Kind, D. (1965). Measuring warehouse space utilization. *Transportation and Distribution Management, 5*(7), 23-33.

Kind, D. (1975). Elements of space utilization. *Transportation and Distribution Management, 15*(2), 29-34.

Kooy, E. (1981). Making better use of available warehouse space. *Industrial Engineering, 13*(10), 26-30.

Matson, J. O., Sonnentag, J. J., White, J. A., & Imhoff, R. C. (2014). *An Analysis of Block Stacking with Lot Splitting*. Paper presented at the IIE Annual Conference. Proceedings.

Matson, J. O., & White, J. A. (1981). *Storage system optimization*. Retrieved from http://goo.gl/FLLLWU

Minitab (Writer). (2010). Minitab 17 Statistical Software.

Montgomery, D. C. (2008). *Design and analysis of experiments*: John Wiley & Sons.

White, J. A., Sonnentag, J. J., & Matson, J. O. (2013). *New Insights Regarding Block Stacking*. Paper presented at the IIE Annual Conference. Proceedings.

CHAPTER SIX: CONCLUSION AND FUTURE RESEARCH

This chapter provides a summary of the overall research approach and a brief narrative of final conclusions. Suggestions for future research directions are included as well.

Contributions

Dense storage provides a means to increase the utilization of available storage space efficiently through stacking pallets horizontally on the floor or vertically in stacks. In this research we study a special class of dense storerooms with an inverted T configuration. A dense storeroom with an inverted T configuration has one horizontal aisle and one vertical aisle.

The primary goal of this dissertation is to study the problem of searching for a specific item or a request of multiple items in such storage environment. The problem of searching arises when we have no information about exact items' storage locations.

The main motivation behind this research is the adoption by the US Navy to have dense storage systems on board of its sea base ships. Meanwhile, such dense storage systems onboard of these ships will remain manually operated for the foreseeable future. Continuous manual storage and retrieval processes of items in a dense storage system, while not keeping track of items' new locations, will allow uncertainty about items' locations to propagate and increase. Uncertainty about items' locations forces the order-picker to search for the item or a group of items before retrieving it.

A thorough review of related existing literature was conducted allowing us to identify gaps. In order to fill such gaps, two main contributions have been made as follows:

Contribution 1

In this contribution we study the search plan problem for a single item in an inverted T k-deep dense storage environment using a single searcher. In an inverted T k-deep system, the searcher can access the most deeply stored pallet without having to reposition more than $k - 1$ items out of his way.

The purpose of this contribution can be summarized as follows:

- Providing a first-time study of the search problem in a warehousing environment, specifically a dense storage environment. Retrieving items in a dense storage environment is challenging because items have to be moved out of the way to gain access to other deeply stored items. Therefore, the search process will have three components: traveling in open aisles, repositioning items, and putting back items that have been repositioned in order to avoid impeding the way to the input/output point.
- Developing a search plan heuristic for a single item in a dense storage environment with an inverted T configuration. Two repositioning policies are studied and a search plan heuristic is developed for both of them. The first repositioning policies assumes items can be repositioned in open aisle locations in the storeroom grid. In such a policy, the items need to be put-back during the search procedure to prevent blocking the way to the input/output point. In the second policy items are repositioned outside the storeroom grid, which requires the availability of extra space, but has the advantage that put-back can be completed after and independent of the search procedure.
- Deriving each component of the search time equation, and use it to calculate the search time for each item in the storeroom, as well as the expected search time and standard deviation for each storeroom configuration.

- Providing insights about the design of searching in very high density storage systems with item location uncertainty through analyzing the results and propose operational search plans, which can be easily articulated to human searchers, for each of the two repositioning policies.

Contribution 2

In this contribution we extend our work by considering that requests are made for multiple items, as well as considering that the storeroom contains items stacked in the vertical direction, not just a single row of items stacked on the floor. Stacking *s* items over each other allows for better utilization of the vertical space, which can increase the number of stored items.

The purpose of this contribution is to:

- Extend our previous study of the search problem in a warehousing environment, specifically a dense storage environment by increasing the number of stored items through stacking pallets on top of each other. This adds complexity when studying searching in a dense storage environment, because items have to be moved from on top of each other, as well as from in front of them, to gain access to other deeply stored items.
- Develop a search plan heuristic for multiple items in a dense storage environment with an inverted T configuration. The repositioning policy that gave better results in the previous chapter is studied and a search plan heuristic is developed for it. The repositioning policy assumes items can be repositioned in open aisle locations in the storeroom grid. In such a policy, the items need to be put-back during the search procedure to prevent blocking the way to the input/output point.

- Derive each component of the search time equation, and use it to calculate the search time for each item in the storeroom, as well as the expected search time and standard deviation for each storeroom configuration.
- Provide insights about the design of searching in very high density storage systems with stacking and searching for multiple items, when item location uncertainty exists. This will be done through analyzing the results and propose search plans to be followed by the searcher.

Conclusions

This dissertation make several contributions from both the scientific and practical perspectives.

From the scientific point of view: We are the first to combine search theory and warehouse operations. We contribute to the warehousing operations literature by considering item location uncertainty. We contribute to the search theory literature by incorporating that items need to be moved out of the way while searching. Also, we develop greedy search heuristics and generate several test problems in order to investigate the heuristics' performance in terms of the expected search time and standard deviation. Test results and the heuristic's performance are numerically and mathematically interesting.

From the practical perspective: Firstly, we provide an efficient solution of how to locate and retrieve goods in an existing manually operated dense storage environments with item location uncertainty. Secondly, our work can be used to equip order pickers in a dense storage systems with either an operational search plan or a search plan that can be generated in seconds using a computer that would lower the overall effort to locate and retrieve an item or a group of items in a storeroom. Finally, our work allows decision makers to incorporate search time into planning activities. For example, our models can aid in understanding search times when a

new dense storeroom is designed, or can be used for downstream planning by knowing the probability distribution function for the effort to search and retrieve an item in a storeroom with a specific known design.

Future Research

While conducting this research, a few directions for future research have been identified:

- In the studied problems we assumed the presence of a single searcher to either search for a single item as in the first contribution or for multiple items in the second contribution. It would be interesting to extend this work by assuming the existence of more than just one searcher in the system.
- We studied a storage environment with a single inverted T configuration, it might be interesting to use the same methodology while assuming having multiple inverted T's not just one.
- We assumed that the searcher had no information about stored item's locations; however, this work can be extended by assuming that the searcher has some sort of a learning curve, i.e. his knowledge about items' locations would increase after each retrieval or after more visits to different locations in the storeroom.
- This work can be extended as well by assuming that the searcher's knowledge about items' storage locations has some kind of probability distribution, i.e. some places within the storeroom has a higher probability of finding the requested item(s) than other places.
- We assumed that every cell in the storeroom in both the vertical and horizontal directions hold a single unique item belongs to different SKU. A possible extension

would be assuming having multiple copies of the same items or items that belong to the same SKU.

- A possible extension could be searching in another warehousing environment that does not have to a dense storage one or even in another dense storage environment with a different configuration. Studying other environments would be a good way to test the heuristic's performance and its robustness.
- From the scientific perspective, it would interesting as well to try to reach an optimal search plan, i.e. reach an optimal solution, and comparing the optimal solution to the solution reached by the heuristic.

APPENDIX A:
CONTRIBUTION 1 EXPERIMENTAL RESULTS

			Repositioning Policy 1		Repositioning Policy 2	
k	***h***	**Density**	**Expected Search Time**	**Standard Deviation**	**Expected Search Time**	**Standard Deviation**
2	6	0.733333	27.45454545	35.37298311	27.81818	35.86418
2	7	0.742857	32	39.95093144	36.23077	46.58682
2	8	0.75	40.93333333	52.05249487	45.13333	58.0481
2	9	0.755556	51.29411765	65.49204236	54.58824	70.42189
2	10	0.76	55.63157895	70.76143056	64.63158	83.78463
2	11	0.763636	67.04761905	86.4335562	75.28571	98.17387
2	12	0.766667	79.60869565	103.1384629	86.56522	113.6095
2	13	0.769231	84.6	110.7768929	98.48	130.1026
2	14	0.771429	98.33333333	129.7233524	111.037	147.6598
2	15	0.773333	113.0689655	149.6473858	124.2414	166.2849
2	16	0.775	119.2903226	160.3761091	138.0968	185.9805
2	17	0.776471	135.2424242	182.5135994	152.6061	206.748
2	18	0.777778	152.1142857	205.6264674	167.7714	228.5887
2	19	0.778947	159.8378378	219.6348337	183.5946	251.503
2	20	0.78	177.9487179	244.9424264	200.0769	275.4915
2	21	0.780952	196.9268293	271.2346559	217.2195	300.5544
2	22	0.781818	206.3023256	288.5711016	235.0233	326.6921
2	23	0.782609	226.5333333	317.047217	253.4889	353.9045
2	24	0.783333	247.5957447	346.5185918	272.617	382.1919
2	25	0.784	258.7142857	367.1888798	292.4082	411.5543
2	26	0.784615	281.0392157	398.8378196	312.8627	441.9916
2	27	0.785185	304.1698113	431.4915656	333.9811	473.5041
2	28	0.785714	317.0909091	455.4883402	355.7636	506.0915
2	29	0.786207	341.4912281	490.3157995	378.2105	539.7541
2	30	0.786667	366.6779661	526.1561262	401.322	574.4918
2	31	0.787097	381.442623	553.4686753	425.0984	610.3045
2	32	0.7875	407.9047619	591.4802312	449.5397	647.1923
2	33	0.787879	435.1384615	630.5113524	474.6462	685.1552
2	34	0.788235	451.7761194	661.1289264	500.4179	724.1932
2	35	0.788571	480.2898551	702.3295729	526.8551	764.3062
2	36	0.788889	509.5633803	744.5553495	553.9577	805.4943
2	37	0.789189	528.0958904	778.4682213	581.726	847.7575
2	38	0.789474	558.6533333	822.8622808	610.16	891.0957
2	39	0.789744	589.961039	868.2861226	639.2597	935.509
2	40	0.79	610.4050633	905.4858269	669.0253	980.9973
2	41	0.790244	643	953.0769922	699.4568	1027.561
2	42	0.790476	676.3373494	1001.701862	730.5542	1075.199
2	43	0.790698	698.7058824	1042.181144	762.3176	1123.912
2	44	0.790909	733.3333333	1092.972559	794.7471	1173.701

2	45	0.791111	768.6966292	1144.801017	827.8427	1224.564
2	46	0.791304	793	1188.553688	861.6044	1276.503
2	47	0.791489	829.655914	1242.548032	896.0323	1329.516
2	48	0.791667	867.0421053	1297.582286	931.1263	1383.605
2	49	0.791837	893.2886598	1344.603065	966.8866	1438.769
2	50	0.792	931.969697	1401.802631	1003.313	1495.007
2	51	0.792157	971.3762376	1460.044587	1040.406	1552.321
2	52	0.792308	999.5728155	1510.328956	1078.165	1610.709
2	53	0.792453	1040.27619	1570.735712	1116.59	1670.173
2	54	0.792593	1081.700935	1632.187024	1155.682	1730.711
2	55	0.792727	1111.853211	1685.731101	1195.44	1792.325
2	56	0.792857	1154.576577	1749.346746	1235.865	1855.014
2	57	0.792982	1198.017699	1814.008852	1276.956	1918.777
2	58	0.793103	1230.130435	1870.809286	1318.713	1983.616
2	59	0.79322	1274.871795	1937.635293	1361.137	2049.529
2	60	0.793333	1320.327731	2005.50945	1404.227	2116.518
2	61	0.793443	1354.404959	2065.563332	1447.983	2184.581
2	62	0.793548	1401.162602	2135.600989	1492.407	2253.72
2	63	0.793651	1448.632	2206.688297	1537.496	2323.934
2	64	0.79375	1484.677165	2269.993092	1583.252	2395.222
2	65	0.793846	1533.449612	2343.243525	1629.674	2467.586
2	66	0.793939	1582.931298	2417.544956	1676.763	2541.024
2	67	0.79403	1620.947368	2484.098441	1724.519	2615.538
2	68	0.794118	1671.733333	2560.562643	1772.941	2691.126
2	69	0.794203	1723.226277	2638.079055	1822.029	2767.789
2	70	0.794286	1763.215827	2707.879275	1871.784	2845.528
2	71	0.794366	1816.014184	2787.558123	1922.206	2924.341
2	72	0.794444	1869.517483	2868.29028	1973.294	3004.23
2	73	0.794521	1911.482759	2941.335505	2025.048	3085.193
2	74	0.794595	1966.292517	3024.229779	2077.469	3167.231
2	75	0.794667	2021.805369	3108.17836	2130.557	3250.345
2	76	0.794737	2065.748344	3184.467054	2184.311	3334.533
2	77	0.794805	2122.568627	3270.57745	2238.732	3419.796
2	78	0.794872	2180.090323	3357.743065	2293.819	3506.134
2	79	0.794937	2226.012739	3437.273858	2349.573	3593.548
2	80	0.795	2284.842767	3526.600999	2405.994	3682.036
2	81	0.795062	2344.372671	3616.984193	2463.081	3771.599
2	82	0.795122	2392.276074	3699.755859	2520.834	3862.237
2	83	0.795181	2453.115152	3792.300304	2579.255	3953.95
2	84	0.795238	2514.652695	3885.901571	2638.341	4046.738
2	85	0.795294	2564.538462	3971.913008	2698.095	4140.602
2	86	0.795349	2627.385965	4067.675264	2758.515	4235.54

2	87	0.795402	2690.930636	4164.495048	2819.601	4331.553
2	88	0.795455	2742.8	4253.745264	2881.354	4428.641
2	89	0.795506	2807.655367	4352.725785	2943.774	4526.804
2	90	0.795556	2873.206704	4452.764491	3006.86	4626.042
2	91	0.795604	2927.060773	4545.252588	3070.613	4726.354
2	92	0.795652	2993.923497	4647.45179	3135.033	4827.742
2	93	0.795699	3061.481081	4750.709784	3200.119	4930.205
2	94	0.795745	3117.320856	4846.434948	3265.872	5033.743
2	95	0.795789	3186.190476	4951.853208	3332.291	5138.356
2	96	0.795833	3255.753927	5058.330826	3399.377	5244.044
2	97	0.795876	3313.580311	5157.292314	3467.13	5350.806
2	98	0.795918	3384.45641	5265.929976	3535.549	5458.644
2	99	0.79596	3456.025381	5375.627524	3604.635	5567.557
2	100	0.796	3515.839196	5477.824662	3674.387	5677.545
3	8	0.803571	92.95555556	103.2451141	108.1111	124.4187
3	9	0.809524	111.7843137	122.8902449	132.3725	151.2125
3	10	0.814286	133.3859649	147.547287	158.1579	179.9272
3	11	0.818182	155.4444444	171.8400676	185.5397	210.7415
3	12	0.821429	179.7246377	199.8464544	214.5652	243.7489
3	13	0.824176	205.24	229.2084257	245.2667	279.0015
3	14	0.826531	232.3333333	260.7822515	277.6667	316.529
3	15	0.828571	260.862069	294.1888975	311.7816	356.3493
3	16	0.830357	291.0645161	329.9946499	347.6237	398.4733
3	17	0.831933	322.7979798	367.804817	385.202	442.9076
3	18	0.833333	356.2571429	408.0801515	424.5238	489.6566
3	19	0.834586	391.3063063	450.443841	465.5946	538.7227
3	20	0.835714	428.1111111	495.2924333	508.4188	590.1077
3	21	0.836735	466.5447154	542.2750367	553	643.8125
3	22	0.837662	506.751938	591.7436465	599.3411	699.8376
3	23	0.838509	548.6148148	643.3736313	647.4444	758.1835
3	24	0.839286	592.2624113	697.4833143	697.3121	818.8502
3	25	0.84	637.585034	753.7723167	748.9456	881.8377
3	26	0.840659	684.6993464	812.5320714	802.3464	947.1461
3	27	0.84127	733.5031447	873.4838932	857.5157	1014.775
3	28	0.841837	784.1030303	936.8969519	914.4545	1084.725
3	29	0.842365	836.4035088	1002.511761	973.1637	1156.995
3	30	0.842857	890.5028249	1070.578602	1033.644	1231.585
3	31	0.843318	946.3114754	1140.85495	1095.896	1308.496
3	32	0.84375	1003.920635	1213.574779	1159.921	1387.727
3	33	0.844156	1063.246154	1288.510582	1225.718	1469.277
3	34	0.844538	1124.373134	1365.882069	1293.289	1553.148
3	35	0.844898	1187.222222	1445.475082	1362.633	1639.338

3	36	0.845238	1251.873239	1527.49673	1433.751	1727.848
3	37	0.84556	1318.251142	1611.744775	1506.644	1818.677
3	38	0.845865	1386.431111	1698.415101	1581.311	1911.825
3	39	0.846154	1456.341991	1787.316163	1657.753	2007.293
3	40	0.846429	1528.054852	1878.633775	1735.97	2105.08
3	41	0.84669	1601.502058	1972.186037	1815.963	2205.187
3	42	0.846939	1676.751004	2068.149675	1897.731	2307.612
3	43	0.847176	1753.737255	2166.35152	1981.275	2412.356
3	44	0.847403	1832.524904	2266.960056	2066.594	2519.419
3	45	0.847619	1913.052434	2369.810057	2153.689	2628.801
3	46	0.847826	1995.380952	2475.062495	2242.56	2740.502
3	47	0.848024	2079.451613	2582.559395	2333.208	2854.522
3	48	0.848214	2165.322807	2692.454857	2425.632	2970.86
3	49	0.848397	2252.938144	2804.597554	2519.832	3089.517
3	50	0.848571	2342.353535	2919.13527	2615.808	3210.492
3	51	0.848739	2433.514851	3035.922793	2713.561	3333.787
3	52	0.848901	2526.475728	3155.102088	2813.091	3459.399
3	53	0.849057	2621.184127	3276.533584	2914.397	3587.331
3	54	0.849206	2717.691589	3400.353865	3017.48	3717.58
3	55	0.849351	2815.948012	3526.428581	3122.339	3850.149
3	56	0.84949	2916.003003	3654.889327	3228.976	3985.035
3	57	0.849624	3017.80826	3785.606599	3337.389	4122.241
3	58	0.849754	3121.411594	3918.707351	3447.58	4261.764
3	59	0.849879	3226.766382	4054.066591	3559.547	4403.606
3	60	0.85	3333.918768	4191.806945	3673.291	4547.766
3	61	0.850117	3442.823691	4331.80763	3788.813	4694.245
3	62	0.85023	3553.525745	4474.187229	3906.111	4843.042
3	63	0.85034	3665.981333	4618.828894	4025.187	4994.157
3	64	0.850446	3780.233596	4765.847422	4146.039	5147.59
3	65	0.850549	3896.24031	4915.129651	4268.669	5303.342
3	66	0.850649	4014.043257	5066.786828	4393.076	5461.412
3	67	0.850746	4133.601504	5220.709251	4519.261	5621.8
3	68	0.85084	4254.955556	5377.004826	4647.222	5784.507
3	69	0.850932	4378.065693	5535.567109	4776.961	5949.531
3	70	0.85102	4502.971223	5696.50086	4908.477	6116.874
3	71	0.851107	4629.63357	5859.702704	5041.771	6286.535
3	72	0.85119	4758.090909	6025.274432	5176.841	6458.514
3	73	0.851272	4888.305747	6193.115566	5313.69	6632.812
3	74	0.851351	5020.315193	6363.325092	5452.315	6809.427
3	75	0.851429	5154.082774	6535.805273	5592.718	6988.361
3	76	0.851504	5289.644592	6710.652438	5734.898	7169.613
3	77	0.851577	5426.965142	6887.771444	5878.856	7353.183

3	78	0.851648	5566.07957	7067.256103	6024.591	7539.071
3	79	0.851718	5706.953291	7249.013733	6172.104	7727.277
3	80	0.851786	5849.620545	7433.135757	6321.394	7917.801
3	81	0.851852	5994.047619	7619.531829	6472.462	8110.644
3	82	0.851916	6140.267894	7808.291101	6625.307	8305.804
3	83	0.851979	6288.248485	7999.325448	6779.929	8503.283
3	84	0.852041	6438.021956	8192.72186	6936.329	8703.08
3	85	0.852101	6589.556213	8388.39433	7094.507	8905.195
3	86	0.852159	6742.883041	8586.427788	7254.462	9109.627
3	87	0.852217	6897.971098	8786.738241	7416.195	9316.378
3	88	0.852273	7054.851429	8989.408656	7579.705	9525.447
3	89	0.852327	7213.493409	9194.356965	7744.992	9736.835
3	90	0.852381	7373.927374	9401.664259	7912.058	9950.54
3	91	0.852433	7536.123389	9611.250306	8080.901	10166.56
3	92	0.852484	7700.111111	9823.194404	8251.521	10384.9
3	93	0.852535	7865.861261	10037.41808	8423.919	10605.56
3	94	0.852584	8033.402852	10253.99892	8598.094	10828.54
3	95	0.852632	8202.707231	10472.86013	8774.048	11053.84
3	96	0.852679	8373.802792	10694.07764	8951.778	11281.45
3	97	0.852725	8546.661485	10917.57629	9131.287	11511.38
3	98	0.85277	8721.311111	11143.43043	9312.573	11743.63
3	99	0.852814	8897.724196	11371.56643	9495.636	11978.2
3	100	0.852857	9075.927973	11602.05714	9680.477	12215.08
4	10	0.844444	227.8947368	229.5043618	285.1053	301.6726
4	11	0.848485	259.8571429	265.677532	334.5238	351.9454
4	12	0.851852	310.1304348	312.887159	387.0217	406.0354
4	13	0.854701	340.4	343.4548005	442.34	463.4312
4	14	0.857143	392.037037	400.1292829	500.5926	524.278
4	15	0.859259	436.7931034	444.074697	561.3793	587.9199
4	16	0.861111	483.0645161	497.9153922	624.9194	654.787
4	17	0.862745	542.530303	556.1008934	691.197	724.8612
4	18	0.864198	582.6285714	603.2716084	760.3143	798.3608
4	19	0.865497	651.7162162	680.3940436	832.2432	875.2158
4	20	0.866667	710.4487179	744.0051893	907.0641	955.5876
4	21	0.867725	769.2682927	811.9840808	984.7439	1039.389
4	22	0.868687	842.2674419	887.3811866	1065.349	1126.75
4	23	0.869565	894.1444444	951.9665734	1148.844	1217.581
4	24	0.87037	978.6808511	1044.799922	1235.287	1311.992
4	25	0.871111	1054.897959	1132.318891	1324.643	1409.894
4	26	0.871795	1126.421569	1215.561001	1416.961	1511.386
4	27	0.872428	1213.660377	1308.708775	1512.208	1616.382
4	28	0.873016	1278.063636	1390.5487	1610.427	1724.973

4	29	0.873563	1376.666667	1497.546348	1711.588	1837.074
4	30	0.874074	1471.949153	1609.621936	1815.729	1952.771
4	31	0.874552	1556.516393	1708.896978	1922.82	2071.985
4	32	0.875	1658.301587	1819.883571	2032.897	2194.793
4	33	0.875421	1735.584615	1918.774436	2145.931	2321.12
4	34	0.875817	1847.455224	2039.57912	2261.955	2451.041
4	35	0.87619	1962.644928	2176.17756	2380.942	2584.483
4	36	0.876543	2060.422535	2291.75295	2502.923	2721.517
4	37	0.876877	2176.910959	2420.580388	2627.87	2862.075
4	38	0.877193	2267.266667	2536.372932	2755.813	3006.221
4	39	0.877493	2391.902597	2670.961358	2886.727	3153.893
4	40	0.877778	2527.5	2831.926636	3020.639	3305.152
4	41	0.878049	2638.580247	2963.90832	3157.525	3459.938
4	42	0.878307	2769.861446	3110.562864	3297.41	3618.308
4	43	0.878553	2873.405882	3243.154153	3440.271	3780.206
4	44	0.878788	3010.471264	3391.595365	3586.132	3945.687
4	45	0.879012	3166.797753	3576.772524	3734.972	4114.696
4	46	0.879227	3291.236264	3725.206303	3886.813	4287.287
4	47	0.879433	3437.365591	3889.673579	4041.634	4463.406
4	48	0.87963	3554.173684	4038.992063	4199.458	4643.106
4	49	0.879819	3703.43299	4201.373581	4360.263	4826.335
4	50	0.88	3880.707071	4410.632938	4524.071	5013.143
4	51	0.880174	4018.539604	4575.5397	4690.861	5203.481
4	52	0.880342	4179.553398	4757.807695	4860.655	5397.396
4	53	0.880503	4309.67619	4923.802463	5033.433	5594.841
4	54	0.880658	4470.957944	5100.20754	5209.215	5795.863
4	55	0.880808	4669.334862	5333.445243	5387.982	6000.416
4	56	0.880952	4820.585586	5514.834727	5569.752	6208.544
4	57	0.881092	4996.50885	5714.89394	5754.509	6420.203
4	58	0.881226	5139.982609	5897.527827	5942.27	6635.436
4	59	0.881356	5313.15812	6088.029359	6133.017	6854.201
4	60	0.881481	5532.752101	6345.162964	6326.769	7076.539
4	61	0.881603	5697.438017	6543.039628	6523.508	7302.41
4	62	0.88172	5888.288618	6760.882642	6723.252	7531.853
4	63	0.881834	6045.14	6960.127847	6925.984	7764.828
4	64	0.881944	6230.110236	7164.787791	7131.72	8001.375
4	65	0.882051	6471.007752	7445.751602	7340.446	8241.455
4	66	0.882155	6649.141221	7660.117311	7552.176	8485.105
4	67	0.882255	6854.932331	7895.738315	7766.895	8732.289
4	68	0.882353	7025.181481	8111.573586	7984.619	8983.043
4	69	0.882448	7221.868613	8330.444087	8205.332	9237.331
4	70	0.88254	7484.136691	8635.185305	8429.05	9495.188

4	71	0.882629	7675.72695	8866.040627	8655.759	9756.579
4	72	0.882716	7896.468531	9119.43503	8885.472	10021.54
4	73	0.882801	8080.131034	9351.843794	9118.176	10290.03
4	74	0.882883	8288.472789	9584.968691	9353.884	10562.09
4	75	0.882963	8572.16443	9913.444454	9592.584	10837.69
4	76	0.883041	8777.218543	10160.7893	9834.288	11116.86
4	77	0.883117	9012.918301	10431.95344	10078.98	11399.56
4	78	0.883191	9210.006452	10680.92256	10326.68	11685.82
4	79	0.883263	9429.952229	10928.33881	10577.38	11975.62
4	80	0.883333	9735.110063	11280.51396	10831.07	12268.99
4	81	0.883402	9953.63354	11544.34794	11087.76	12565.9
4	82	0.883469	10204.29755	11833.27885	11347.45	12866.37
4	83	0.883534	10414.82121	12098.79773	11610.14	13170.37
4	84	0.883598	10646.32934	12360.53666	11875.83	13477.94
4	85	0.88366	10972.98817	12736.38206	12144.51	13789.05
4	86	0.883721	11204.98538	13016.70463	12416.2	14103.73
4	87	0.88378	11470.6185	13323.3999	12690.88	14421.93
4	88	0.883838	11694.58571	13605.45991	12968.56	14743.71
4	89	0.883895	11937.62147	13881.54818	13249.24	15069.02
4	90	0.883951	12285.81006	14281.03947	13532.92	15397.9
4	91	0.884005	12531.28453	14577.85004	13819.6	15730.31
4	92	0.884058	12811.89071	14902.30769	14109.28	16066.28
4	93	0.88411	13049.30811	15200.90168	14401.95	16405.8
4	94	0.884161	13303.84225	15491.36217	14697.63	16748.87
4	95	0.884211	13673.58466	15914.47878	14996.3	17095.49
4	96	0.884259	13932.53927	16227.77678	15297.97	17445.67
4	97	0.884307	14228.12176	16569.99514	15602.64	17799.38
4	98	0.884354	14478.99487	16885.11715	15910.31	18156.66
4	99	0.8844	14745.00254	17189.96957	16220.98	18517.48
4	100	0.884444	15136.3191	17636.694	16534.65	18881.86
5	12	0.871212	444.426087	425.3305428	587.0522	586.9519
5	13	0.874126	512.088	490.3764271	672.2	667.8444
5	14	0.876623	565.7111111	542.3660765	758.037	749.8856
5	15	0.878788	636.6413793	609.8697959	847.5517	836.117
5	16	0.880682	710.0709677	682.5933197	940.8581	926.7863
5	17	0.882353	771.9333333	746.6102722	1038.042	1021.98
5	18	0.883838	853.72	826.8895635	1139.171	1121.822
5	19	0.885167	939.1513514	914.0002004	1244.297	1226.334
5	20	0.886364	1011.287179	992.485892	1353.462	1335.586
5	21	0.887446	1104.580488	1086.433354	1466.698	1449.575
5	22	0.88843	1202.395349	1187.96192	1584.033	1568.344
5	23	0.889328	1285.666667	1281.34632	1705.489	1691.881

5	24	0.890152	1390.940426	1389.449331	1831.085	1820.216
5	25	0.890909	1501.359184	1505.406163	1960.837	1953.333
5	26	0.891608	1596.262745	1613.739925	2094.757	2091.254
5	27	0.892256	1713.784906	1736.16956	2232.857	2233.964
5	28	0.892857	1836.92	1866.518484	2375.145	2381.478
5	29	0.893417	1943.764912	1989.743641	2521.632	2533.782
5	30	0.893939	2073.698305	2126.556145	2672.322	2690.889
5	31	0.894428	2209.609836	2271.260531	2827.223	2852.787
5	32	0.894886	2328.6	2409.296388	2986.34	3019.485
5	33	0.895317	2471.049231	2560.504932	3149.677	3190.973
5	34	0.895722	2619.770149	2719.536883	3317.239	3367.258
5	35	0.896104	2751.046377	2872.306646	3489.029	3548.331
5	36	0.896465	2906.08169	3037.908449	3665.051	3734.2
5	37	0.896806	3067.630137	3211.248648	3845.307	3924.854
5	38	0.897129	3211.293333	3378.686376	4029.8	4120.301
5	39	0.897436	3378.963636	3558.673322	4218.532	4320.533
5	40	0.897727	3553.349367	3746.309034	4411.506	4525.555
5	41	0.898004	3709.474074	3928.360127	4608.723	4735.36
5	42	0.898268	3889.814458	4122.722963	4810.186	4949.953
5	43	0.89852	4077.042353	4324.646118	5015.894	5169.328
5	44	0.89876	4245.685057	4521.265909	5225.851	5393.489
5	45	0.89899	4438.721348	4729.99584	5440.056	5622.431
5	46	0.899209	4638.793407	4946.20156	5658.512	5856.158
5	47	0.89942	4819.997849	5157.353558	5881.219	6094.664
5	48	0.899621	5025.749474	5380.442781	6108.179	6337.954
5	49	0.899814	5238.665979	5610.928346	6339.392	6586.021
5	50	0.9	5432.466667	5836.582637	6574.859	6838.872
5	51	0.900178	5650.948515	6074.024434	6814.58	7096.499
5	52	0.90035	5876.708738	6318.788562	7058.557	7358.908
5	53	0.900515	6083.133333	6558.920477	7306.79	7626.093
5	54	0.900673	6314.357009	6810.709144	7559.28	7898.059
5	55	0.900826	6552.959633	7069.751509	7816.028	8174.801
5	56	0.900974	6772.030631	7324.340559	8077.032	8456.322
5	57	0.901116	7016.00531	7590.47128	8342.296	8742.618
5	58	0.901254	7267.448696	7863.792179	8611.817	9033.693
5	59	0.901387	7499.184615	8132.821228	8885.598	9329.543
5	60	0.901515	7755.917647	8413.289939	9163.639	9630.171
5	61	0.901639	8020.2	8700.89007	9445.939	9935.573
5	62	0.90176	8264.61626	8984.344676	9732.499	10245.75
5	63	0.901876	8534.1136	9279.147944	10023.32	10560.71
5	64	0.901989	8811.233071	9581.028252	10318.4	10880.44
5	65	0.902098	9068.342636	9878.896161	10617.74	11204.94

5	66	0.902204	9350.60916	10188.03108	10921.35	11534.22
5	67	0.902307	9640.56391	10504.19265	11229.21	11868.27
5	68	0.902406	9910.377778	10816.46339	11541.34	12207.1
5	69	0.902503	10205.41752	11139.92749	11857.73	12550.71
5	70	0.902597	10508.20576	11470.37149	12178.38	12899.08
5	71	0.902689	10790.73333	11797.03607	12503.3	13252.23
5	72	0.902778	11098.54965	12134.82722	12832.47	13610.16
5	73	0.902864	11414.16966	12479.55486	13165.91	13972.86
5	74	0.902948	11709.41905	12820.60547	13503.61	14340.33
5	75	0.90303	12030.01477	13172.72188	13845.58	14712.58
5	76	0.90311	12358.4649	13531.73433	14191.81	15089.6
5	77	0.903188	12666.44314	13887.16421	14542.3	15471.39
5	78	0.903263	12999.82065	14253.60431	14897.05	15857.95
5	79	0.903337	13341.09936	14626.90276	15256.07	16249.29
5	80	0.903409	13661.81258	14996.70597	15619.35	16645.4
5	81	0.903479	14007.97391	15377.46841	15986.9	17046.29
5	82	0.903548	14362.07975	15765.05398	16358.71	17451.94
5	83	0.903614	14695.53333	16149.22532	16734.78	17862.37
5	84	0.90368	15054.48024	16544.30893	17115.12	18277.58
5	85	0.903743	15421.41183	16946.18271	17499.72	18697.55
5	86	0.903805	15767.61053	17344.71757	17888.59	19122.3
5	87	0.903866	16139.34451	17754.12133	18281.72	19551.82
5	88	0.903926	16519.10057	18170.28436	18679.11	19986.11
5	89	0.903984	16878.04859	18583.17867	19080.77	20425.17
5	90	0.90404	17262.57095	19006.90168	19486.7	20869.01
5	91	0.904096	17655.15028	19437.35495	19896.89	21317.62
5	92	0.90415	18026.85137	19864.60506	20311.34	21771
5	93	0.904203	18424.16324	20302.64654	20730.06	22229.15
5	94	0.904255	18829.56471	20747.39098	21153.04	22692.07
5	95	0.904306	19214.02222	21188.99364	21580.29	23159.77
5	96	0.904356	19624.12461	21641.3529	22011.8	23632.24
5	97	0.904405	20042.34715	22100.3894	22447.58	24109.48
5	98	0.904453	20439.5641	22556.3417	22887.63	24591.49
5	99	0.9045	20862.45787	23023.01811	23331.93	25078.27
5	100	0.904545	21293.5005	23496.34752	23780.51	25569.82
6	14	0.89011	782.5555556	721.4162169	1046.395	1004.316
6	15	0.892308	852.8275862	781.6984706	1170.057	1117.89
6	16	0.894231	947.8817204	871.2895134	1298.097	1235.847
6	17	0.895928	1059.727273	975.6129575	1430.727	1358.708
6	18	0.897436	1152.085714	1069.759289	1568.305	1487.115
6	19	0.898785	1250.801802	1170.611504	1710.748	1620.929
6	20	0.9	1344.982906	1255.784122	1858.333	1760.604

6	21	0.901099	1481.243902	1383.163614	2010.959	1905.928
6	22	0.902098	1584.069767	1495.88991	2168.853	2057.265
6	23	0.90301	1706.985185	1611.133629	2331.904	2214.375
6	24	0.903846	1863.815603	1767.998408	2500.305	2377.571
6	25	0.904615	1981.673469	1888.930135	2673.946	2546.608
6	26	0.905325	2099.313725	2004.955705	2852.993	2721.764
6	27	0.905983	2239.477987	2161.835811	3037.34	2902.798
6	28	0.906593	2415.436364	2330.412988	3227.133	3089.963
6	29	0.907162	2552.216374	2482.710146	3422.269	3283.029
6	30	0.907692	2707.175141	2638.023218	3622.881	3482.228
6	31	0.908189	2906.73224	2835.762074	3828.869	3687.339
6	32	0.908654	3057.873016	3000.195348	4040.354	3898.581
6	33	0.909091	3197.333333	3143.966655	4257.241	4115.742
6	34	0.909502	3388.885572	3361.87557	4479.642	4339.029
6	35	0.90989	3605.024155	3573.290012	4707.464	4568.24
6	36	0.910256	3778.849765	3769.77528	4940.812	4803.569
6	37	0.910603	3966.739726	3963.256241	5179.598	5044.824
6	38	0.910931	4204.915556	4195.96954	5423.92	5292.192
6	39	0.911243	4392.645022	4405.060416	5673.693	5545.487
6	40	0.911538	4558.244726	4585.263787	5929.008	5804.888
6	41	0.91182	4799.407407	4855.329454	6189.786	6070.217
6	42	0.912088	5058.903614	5114.441482	6456.112	6341.646
6	43	0.912343	5271.662745	5356.257975	6727.91	6619.003
6	44	0.912587	5491.881226	5586.028067	7005.261	6902.454
6	45	0.912821	5766.28839	5852.304946	7288.09	7191.835
6	46	0.913043	5991.040293	6104.144553	7576.476	7487.304
6	47	0.913257	6186.261649	6325.039002	7870.348	7788.703
6	48	0.913462	6478.336842	6648.201156	8169.779	8096.185
6	49	0.913658	6779.580756	6950.615373	8474.701	8409.597
6	50	0.913846	7033.239057	7239.621524	8785.185	8729.088
6	51	0.914027	7285.363036	7505.016215	9101.165	9054.51
6	52	0.914201	7594.533981	7804.711937	9422.709	9386.005
6	53	0.914369	7856.463492	8098.336549	9749.752	9723.433
6	54	0.91453	8083.392523	8361.256249	10082.36	10066.93
6	55	0.914685	8426.987768	8737.530146	10420.47	10416.36
6	56	0.914835	8769.918919	9083.27358	10764.15	10771.86
6	57	0.91498	9064.864307	9418.489249	11113.34	11133.29
6	58	0.915119	9348.602899	9719.280328	11468.09	11500.78
6	59	0.915254	9691.57265	10052.58562	11828.36	11874.2
6	60	0.915385	9990.747899	10387.50364	12194.19	12253.69
6	61	0.915511	10250.71625	10692.78198	12565.53	12639.11
6	62	0.915633	10646.21409	11122.09695	12942.44	13030.58

6	63	0.915751	11030.05867	11509.80643	13324.87	13428
6	64	0.915865	11367.25722	11892.08382	13712.87	13831.46
6	65	0.915976	11682.4031	12228.20492	14106.38	14240.86
6	66	0.916084	12058.50382	12595.37757	14505.47	14656.31
6	67	0.916188	12394.95489	12971.32176	14910.07	15077.7
6	68	0.91629	12688.8642	13318.94727	15320.24	15505.14
6	69	0.916388	13136.52068	13801.21912	15735.93	15938.52
6	70	0.916484	13561.3789	14231.5495	16157.19	16377.94
6	71	0.916576	13940.85106	14659.93794	16583.98	16823.3
6	72	0.916667	14287.25175	15031.38535	17016.33	17274.71
6	73	0.916754	14696.0023	15432.67149	17454.21	17732.05
6	74	0.91684	15069.7415	15849.48894	17897.66	18195.44
6	75	0.916923	15398.23043	16239.33551	18346.63	18664.77
6	76	0.917004	15898.22517	16774.48668	18801.18	19140.13
6	77	0.917083	16363.57734	17246.61786	19261.25	19621.44
6	78	0.91716	16785.92258	17721.75529	19726.89	20108.79
6	79	0.917235	17163.46285	18128.54917	20198.05	20602.08
6	80	0.917308	17604.50524	18564.16319	20674.79	21101.41
6	81	0.917379	18015.53623	19021.76271	21157.06	21606.67
6	82	0.917448	18379.07362	19453.67485	21644.89	22117.98
6	83	0.917516	18931.53737	20041.63901	22138.26	22635.23
6	84	0.917582	19437.53094	20556.24244	22637.19	23158.51
6	85	0.917647	19902.6568	21077.33944	23141.66	23687.73
6	86	0.91771	20311.24756	21519.50789	23651.69	24222.99
6	87	0.917772	20784.30829	21989.6307	24167.26	24764.19
6	88	0.917832	21232.63048	22487.95615	24688.39	25311.42
6	89	0.917891	21631.57062	22961.78087	25215.06	25864.6
6	90	0.917949	22236.60149	23602.5026	25747.3	26423.81
6	91	0.918005	22782.85083	24158.95518	26285.07	26988.97
6	92	0.91806	23291.18215	24726.55524	26828.41	27560.15
6	93	0.918114	23730.75315	25204.1276	27377.28	28137.28
6	94	0.918167	24235.61854	25708.91042	27931.73	28720.44
6	95	0.918219	24721.22928	26247.92635	28491.7	29309.54
6	96	0.918269	25155.84642	26763.52418	29057.25	29904.68
6	97	0.918319	25813.51986	27456.95753	29628.33	30505.76
6	98	0.918367	26400.18291	28055.86179	30204.98	31112.86
6	99	0.918415	26951.59052	28669.30674	30787.16	31725.91
6	100	0.918462	27422.08543	29182.31087	31374.92	32344.99
7	16	0.904167	1237.138249	1107.042566	1701.166	1580.64
7	17	0.905882	1330.160173	1194.739985	1874.056	1734.882
7	18	0.907407	1475.391837	1318.292124	2052.829	1895.408
7	19	0.908772	1588.467181	1433.278563	2237.734	2062.585

7	20	0.91	1738.186813	1558.418499	2428.897	2236.685
7	21	0.911111	1871.696864	1702.376808	2626.519	2417.901
7	22	0.912121	2021.41196	1822.513867	2830.548	2606.195
7	23	0.913043	2172.853968	1988.035183	3041.146	2801.708
7	24	0.913889	2326.878419	2113.339767	3256.939	3001.684
7	25	0.914667	2499.297376	2302.988212	3479.752	3209.747
7	26	0.915385	2656.232493	2433.088109	3707.874	3422.684
7	27	0.916049	2830.67655	2616.574562	3942.978	3643.603
7	28	0.916667	3011.561039	2784.799034	4183.816	3870.401
7	29	0.917241	3194.924812	2973.90008	4431.256	4104.568
7	30	0.917778	3389.658596	3161.046737	4685.189	4345.874
7	31	0.91828	3584.016393	3360.595567	4945.782	4594.561
7	32	0.91875	3795.072562	3570.518502	5212.927	4850.404
7	33	0.919192	3999.826374	3779.58362	5486.776	5113.631
7	34	0.919608	4224.820896	4006.143431	5767.222	5384.02
7	35	0.92	4441.356108	4227.967517	6054.404	5661.79
7	36	0.92037	4682.114688	4474.317971	6348.219	5946.726
7	37	0.920721	4909.551859	4707.041324	6648.796	6239.035
7	38	0.921053	5164.390476	4968.930387	6956.036	6538.51
7	39	0.921368	5403.827458	5215.008579	7270.058	6845.35
7	40	0.921667	5674.435805	5495.726121	7590.765	7159.356
7	41	0.921951	5925.303351	5753.8491	7918.27	7480.718
7	42	0.922222	6209.901893	6049.135323	8252.48	7809.243
7	43	0.922481	6473.248739	6321.814115	8593.501	8145.115
7	44	0.922727	6773.259442	6634.387546	8941.243	8488.15
7	45	0.922963	7047.256822	6917.418343	9295.806	8838.522
7	46	0.923188	7362.350078	7246.384829	9657.104	9196.055
7	47	0.923404	7648.490015	7543.149622	10025.23	9560.918
7	48	0.923611	7979.396992	7889.929738	10400.1	9932.94
7	49	0.92381	8277.450663	8199.859234	10781.81	10312.28
7	50	0.924	8622.408369	8560.33065	11170.28	10698.79
7	51	0.924183	8933.755304	8886.458476	11565.59	11092.6
7	52	0.924359	9293.407767	9262.027915	11967.66	11493.58
7	53	0.924528	9616.921088	9601.584129	12376.58	11901.86
7	54	0.924691	9990.548732	9990.680158	12792.27	12317.29
7	55	0.924848	10327.02359	10345.31324	13214.81	12740.03
7	56	0.925	10715.68983	10750.4156	13644.12	13169.93
7	57	0.925146	11064.64602	11118.87273	14080.3	13607.12
7	58	0.925287	11467.1118	11537.19644	14523.25	14051.46
7	59	0.925424	11829.22466	11921.02248	14973.07	14503.1
7	60	0.925556	12246.53421	12354.87887	15429.67	14961.89
7	61	0.925683	12620.41204	12750.75854	15893.13	15427.97

7	62	0.925806	13052.34959	13199.69004	16363.38	15901.2
7	63	0.925926	13439.02057	13609.95861	16840.49	16381.71
7	64	0.926042	13886.15861	14075.24698	17324.41	16869.37
7	65	0.926154	14285.40975	14499.35479	17815.18	17364.32
7	66	0.926263	14746.45256	14978.00972	18312.75	17866.42
7	67	0.926368	15159.28357	15418.18846	18817.19	18375.79
7	68	0.926471	15634.7291	15911.38341	19328.43	18892.31
7	69	0.92657	16060.25965	16365.46369	19846.54	19416.11
7	70	0.926667	16549.56732	16872.03427	20371.45	19947.06
7	71	0.926761	16988.37589	17341.24566	20903.23	20485.28
7	72	0.926852	17492.37463	17863.17849	21441.82	21030.65
7	73	0.926941	17944.06995	18346.51276	21987.27	21583.29
7	74	0.927027	18461.80855	18881.666	22539.54	22143.08
7	75	0.927111	18926.89453	19380.28531	23098.67	22710.13
7	76	0.927193	19459.19678	19930.54351	23664.62	23284.34
7	77	0.927273	19936.56116	20441.79078	24237.44	23865.8
7	78	0.92735	20483.26728	21006.82558	24817.07	24454.42
7	79	0.927426	20973.69882	21532.51591	25403.57	25050.3
7	80	0.9275	21535.27673	22113.40615	25996.89	25653.33
7	81	0.927572	22038.59184	22653.07754	26597.07	26263.62
7	82	0.927642	22614.01665	23247.44815	27204.08	26881.06
7	83	0.927711	23131.00346	23802.88966	27817.95	27505.76
7	84	0.927778	23720.68007	24411.70706	28438.65	28137.61
7	85	0.927843	24250.6213	24981.16316	29066.21	28776.71
7	86	0.927907	24854.11612	25603.48022	29700.6	29422.96
7	87	0.927969	25397.46903	26187.94792	30341.85	30076.47
7	88	0.92803	26015.46041	26825.39713	30989.93	30737.13
7	89	0.92809	26571.89911	27424.05086	31644.88	31405.04
7	90	0.928148	27203.61453	28074.87739	32306.66	32080.11
7	91	0.928205	27773.54301	28688.65882	32975.3	32762.42
7	92	0.928261	28419.66198	29354.43541	33650.77	33451.88
7	93	0.928315	29002.15676	29981.14041	34333.11	34148.6
7	94	0.928369	29662.55233	30661.60254	35022.28	34852.47
7	95	0.928421	30258.25472	31302.72153	35718.32	35563.58
7	96	0.928472	30933.32162	31998.78758	36421.19	36281.85
7	97	0.928522	31542.07328	32653.93012	37130.92	37007.36
7	98	0.928571	32230.96337	33363.62439	37847.49	37740.03
7	99	0.92862	32853.41624	34034.28785	38570.92	38479.94
7	100	0.928667	33556.47021	34758.42464	39301.19	39227.01
8	18	0.915033	1815.707143	1596.327813	2585.9	2344.904
8	19	0.916409	1976.398649	1727.998926	2816.865	2546.486
8	20	0.917647	2149.852564	1875.900199	3054.942	2755.579

8	21	0.918768	2300.884146	2027.335136	3300.348	2972.595
8	22	0.919786	2500.959302	2192.961086	3553.523	3198.152
8	23	0.920716	2668.505556	2352.268761	3813.456	3430.276
8	24	0.921569	2852.473404	2511.526157	4081.314	3671.037
8	25	0.922353	3036.19898	2694.868215	4354.526	3915.626
8	26	0.923077	3255.95098	2895.994696	4636.363	4170.209
8	27	0.923747	3460.622642	3100.278858	4923.717	4429.359
8	28	0.92437	3663.931818	3277.849357	5219.605	4698.326
8	29	0.924949	3890.964912	3489.3401	5521.75	4973.542
8	30	0.92549	4110.838983	3726.026522	5831.72	5257.539
8	31	0.925996	4366.491803	3956.470364	6149.303	5549.871
8	32	0.926471	4604.638889	4194.63025	6474.79	5850.981
8	33	0.926916	4849.807692	4437.315105	6807.973	6160.438
8	34	0.927336	5093.835821	4684.3999	7149.119	6478.662
8	35	0.927731	5380.605072	4954.614419	7498.029	6805.235
8	36	0.928105	5652.28169	5234.262431	7854.947	7140.559
8	37	0.928458	5916.753425	5491.863684	8219.682	7484.231
8	38	0.928793	6207.256667	5772.922488	8592.46	7836.634
8	39	0.92911	6500.899351	6086.636737	8973.097	8197.384
8	40	0.929412	6832	6407.635813	9361.807	8566.845
8	41	0.929699	7138.888889	6719.706287	9758.41	8944.649
8	42	0.929972	7452.551205	7039.2446	10163.11	9331.146
8	43	0.930233	7765.314706	7366.92265	10575.73	9725.98
8	44	0.930481	8121.267241	7709.061578	10996.46	10129.49
8	45	0.930719	8462.794944	8062.39524	11425.14	10541.34
8	46	0.930946	8798.942308	8408.093706	11861.95	10961.84
8	47	0.931164	9151.669355	8755.069217	12306.73	11390.68
8	48	0.931373	9515.460526	9137.874514	12759.64	11828.16
8	49	0.931573	9931.056701	9554.304771	13220.54	12273.96
8	50	0.931765	10302.10859	9930.390831	13689.59	12728.4
8	51	0.931949	10682.34158	10320.57214	14166.63	13191.17
8	52	0.932127	11077.10194	10745.00976	14651.84	13662.55
8	53	0.932297	11497.5881	11150.4397	15145.05	14142.25
8	54	0.932462	11908.50935	11575.14067	15646.43	14630.57
8	55	0.93262	12322	12011.63752	16155.84	15127.2
8	56	0.932773	12740.44144	12430.56556	16673.41	15632.43
8	57	0.932921	13171.05973	12877.48897	17199.02	16145.98
8	58	0.933063	13677.66304	13393.79345	17732.8	16668.12
8	59	0.9332	14108.38675	13828.38065	18274.62	17198.57
8	60	0.933333	14558.09244	14293.68664	18824.62	17737.6
8	61	0.933462	15039.44215	14814.67779	19382.68	18284.96
8	62	0.933586	15520.03252	15278.31548	19948.91	18840.88

8	63	0.933707	16002.066	15777.57199	20523.2	19405.12
8	64	0.933824	16495.72638	16303.55205	21105.67	19977.92
8	65	0.933937	16980.74806	16794.86516	21696.2	20559.04
8	66	0.934046	17478.39122	17308.87371	22294.92	21148.71
8	67	0.934153	18073.66917	17915.00354	22901.71	21746.7
8	68	0.934256	18567.29259	18417.10278	23516.69	22353.24
8	69	0.934356	19084.87774	18953.27154	24139.74	22968.1
8	70	0.934454	19652.93885	19565.30713	24770.98	23591.51
8	71	0.934548	20196.58865	20095.92702	25410.3	24223.22
8	72	0.934641	20747.77448	20665.56034	26057.81	24863.48
8	73	0.93473	21324.14655	21281.31185	26713.4	25512.06
8	74	0.934817	21876.25	21845.08931	27377.18	26169.17
8	75	0.934902	22439.87584	22425.8624	28049.05	26834.59
8	76	0.934985	23124.22351	23120.45025	28729.11	27508.55
8	77	0.935065	23682.08824	23692.82068	29417.27	28190.82
8	78	0.935143	24265.50968	24295.76301	30113.61	28881.62
8	79	0.93522	24921.93949	24999.45301	30818.05	29580.74
8	80	0.935294	25529.89151	25599.91601	31530.68	30288.38
8	81	0.935367	26147.96739	26235.94885	32251.41	31004.34
8	82	0.935438	26809.51534	26943.2382	32980.33	31728.81
8	83	0.935507	27429.01061	27579.45122	33717.35	32461.61
8	84	0.935574	28057.11527	28225.07627	34462.56	33202.91
8	85	0.93564	28831.53994	29009.28548	35215.87	33952.54
8	86	0.935705	29454.54532	29653.19838	35977.38	34710.68
8	87	0.935767	30104.08815	30324.52999	36747	35477.14
8	88	0.935829	30848.28714	31116.47877	37524.8	36252.11
8	89	0.935889	31521.39124	31788.16864	38310.72	37035.4
8	90	0.935948	32206.4148	32492.14711	39104.83	37827.19
8	91	0.936005	32953.19061	33288.21952	39907.04	38627.3
8	92	0.936061	33640.28142	33996.81185	40717.45	39435.92
8	93	0.936116	34333.38649	34709.72497	41535.98	40252.86
8	94	0.93617	35196.98663	35580.79591	42362.69	41078.3
8	95	0.936223	35885.76058	36296.84918	43197.52	41912.06
8	96	0.936275	36600.74476	37035.95668	44040.54	42754.32
8	97	0.936325	37433.14378	37915.82305	44891.68	43604.9
8	98	0.936375	38172.00256	38659.42749	45751.01	44463.98
8	99	0.936423	38923.1599	39430.77678	46618.46	45331.38
8	100	0.936471	39756.03894	40315.50612	47494.1	46207.28
9	20	0.923684	2600.361823	2239.446959	3741.604	3330.055
9	21	0.924812	2781.823848	2398.071567	4039.461	3584.506
9	22	0.925837	2991.795866	2578.954583	4346.84	3849.506
9	23	0.926773	3222.209877	2775.725978	4661.928	4121.723

9	24	0.927632	3409.066194	2969.57526	4982.021	4396.89
9	25	0.928421	3656.034014	3157.585171	5312.315	4683.594
9	26	0.92915	3885.309368	3377.516004	5647.85	4974.427
9	27	0.929825	4113.851153	3584.927763	5993.419	5276.567
9	28	0.930451	4398.127273	3840.128481	6345.356	5585.186
9	29	0.931034	4632.224172	4094.262382	6706.096	5903.675
9	30	0.931579	4890.706215	4287.390869	7075.407	6231.571
9	31	0.932088	5165.735883	4557.983826	7453.67	6569.411
9	32	0.932566	5444.061728	4833.841865	7840.654	6916.732
9	33	0.933014	5773.509402	5132.797943	8236.703	7274.03
9	34	0.933437	6038.323383	5412.34473	8641.587	7640.85
9	35	0.933835	6330.793881	5658.146324	9055.622	8017.658
9	36	0.934211	6651.231612	5977.195455	9478.584	8404.01
9	37	0.934566	6977.961948	6306.333323	9910.763	8800.347
9	38	0.934903	7352.632593	6653.990256	10351.94	9206.238
9	39	0.935223	7656.258297	6974.772347	10802.39	9622.105
9	40	0.935526	7989.888889	7279.448046	11261.9	10047.53
9	41	0.935815	8353.576132	7641.341158	11730.72	10482.91
9	42	0.93609	8726.978581	8019.162734	12208.65	10927.86
9	43	0.936353	9149.248366	8417.07913	12695.92	11382.74
9	44	0.936603	9493.375479	8781.662851	13192.35	11847.18
9	45	0.936842	9872.84769	9147.13512	13698.14	12321.55
9	46	0.937071	10279.06593	9550.884188	14213.12	12805.48
9	47	0.93729	10699.50657	9977.303526	14737.49	13299.31
9	48	0.9375	11170.9462	10426.51137	15271.08	13802.69
9	49	0.937701	11557.8866	10839.00202	15814.07	14315.97
9	50	0.937895	11983.71156	11261.16332	16366.31	14838.79
9	51	0.93808	12430.85039	11704.18415	16927.97	15371.49
9	52	0.938259	12900.14563	12182.02719	17498.89	15913.73
9	53	0.938431	13420.55767	12680.91322	18079.25	16465.84
9	54	0.938596	13851.06231	13141.55948	18668.89	17027.48
9	55	0.938756	14323.36086	13617.18722	19267.97	17598.98
9	56	0.93891	14811.77477	14100.95842	19876.35	18180.01
9	57	0.939058	15331.90462	14633.07055	20494.19	18770.88
9	58	0.939201	15900.77391	15180.20892	21121.33	19371.28
9	59	0.93934	16375.50332	15689.17748	21757.95	19981.51
9	60	0.939474	16893.52661	16214.57877	22403.88	20601.27
9	61	0.939603	17424.4472	16742.67026	23059.28	21230.86
9	62	0.939728	17995.16802	17326.92244	23724.01	21869.96
9	63	0.93985	18613.10667	17923.44835	24398.22	22518.88
9	64	0.939967	19130.93526	18477.51188	25081.77	23177.32
9	65	0.940081	19696.44186	19055.52246	25774.81	23845.56

9	66	0.940191	20269.33927	19626.49444	26477.19	24523.33
9	67	0.940299	20891.61988	20264.2918	27189.06	25210.88
9	68	0.940402	21558.64938	20909.91265	27910.28	25907.96
9	69	0.940503	22120.06083	21509.85956	28641	26614.83
9	70	0.940602	22731.77378	22136.43463	29381.07	27331.21
9	71	0.940697	23350.08432	22757.48305	30130.64	28057.38
9	72	0.940789	24022.57265	23445.37774	30889.58	28793.06
9	73	0.94088	24738.19387	24139.02353	31658.02	29538.52
9	74	0.940967	25342.24414	24782.65867	32435.83	30293.49
9	75	0.941053	26001.74422	25460.53279	33223.13	31058.23
9	76	0.941136	26664.92642	26129.71694	34019.82	31832.49
9	77	0.941217	27387.67248	26867.63862	34826.01	32616.51
9	78	0.941296	28152.3276	27610.3202	35641.58	33410.05
9	79	0.941372	28799.47983	28298.74813	36466.65	34213.34
9	80	0.941447	29507.32006	29028.33412	37301.11	35026.15
9	81	0.94152	30214.14976	29742.61392	38145.08	35848.71
9	82	0.941592	30987.80709	30531.84404	38998.43	36680.79
9	83	0.941661	31801.49562	31323.43555	39861.29	37522.62
9	84	0.941729	32490.96407	32054.99608	40733.55	38373.96
9	85	0.941796	33247.89283	32837.11794	41615.31	39235.04
9	86	0.94186	33998.18259	33596.06382	42506.47	40105.65
9	87	0.941924	34823.74117	34438.43328	43407.14	40985.99
9	88	0.941986	35686.04127	35278.07577	44317.21	41875.84
9	89	0.942046	36418.26428	36053.92332	45236.79	42775.44
9	90	0.942105	37224.61887	36888.5648	46165.77	43684.55
9	91	0.942163	38017.82873	37691.22051	47104.26	44603.39
9	92	0.94222	38894.966	38585.44359	48052.16	45531.75
9	93	0.942275	39806.23363	39474.0042	49009.57	46469.83
9	94	0.942329	40580.54011	40292.77718	49976.39	47417.44
9	95	0.942382	41436.86714	41180.42389	50952.71	48374.76
9	96	0.942434	42272.80454	42026.57082	51938.45	49341.61
9	97	0.942485	43202.05354	42973.61157	52933.7	50318.18
9	98	0.942535	44162.28661	43911.02873	53938.36	51304.27
9	99	0.942584	44979.08911	44773.84271	54952.53	52300.07
9	100	0.942632	45885.59855	45714.3128	55976.12	53305.4
10	22	0.930736	3546.539535	3020.565213	5176.433	4526.48
10	23	0.931677	3779.675556	3211.331316	5546.431	4837.883
10	24	0.93254	4013.025532	3406.103854	5928.97	5162.936
10	25	0.933333	4286.653061	3671.172935	6316.739	5490.908
10	26	0.934066	4581.117647	3906.324566	6716.686	5831.901
10	27	0.934744	4822.909434	4118.666725	7121.875	6176.923
10	28	0.935374	5158.458182	4428.396115	7538.778	6534.47

10	29	0.935961	5430.540351	4656.047913	7962.221	6898.677
10	30	0.936508	5710.789831	4911.737907	8395.488	7273.64
10	31	0.93702	6037.881967	5211.134148	8838.269	7658.791
10	32	0.9375	6359.647619	5503.562018	9291.079	8054.904
10	33	0.937951	6731.830769	5830.279026	9753.609	8461.396
10	34	0.938375	7029.131343	6130.786748	10226.33	8878.973
10	35	0.938776	7357.657971	6438.231202	10708.92	9307.05
10	36	0.939153	7691.185915	6759.325285	11201.83	9746.284
10	37	0.939511	8093.775342	7112.651355	11704.74	10196.09
10	38	0.93985	8437.930667	7462.061479	12218.05	10657.1
10	39	0.940171	8873.215584	7855.935923	12741.48	11128.73
10	40	0.940476	9253.453165	8226.112449	13275.38	11611.58
10	41	0.940767	9643.012346	8601.734581	13819.47	12105.08
10	42	0.941043	10065.83133	9009.196076	14374.11	12609.81
10	43	0.941307	10513.23059	9453.823738	14939.01	13125.21
10	44	0.941558	10968.18161	9851.180811	15514.49	13651.82
10	45	0.941799	11405.58876	10329.62465	16100.3	14189.13
10	46	0.942029	11810.42418	10711.85459	16696.73	14737.64
10	47	0.942249	12255.86452	11168.80566	17303.54	15296.84
10	48	0.94246	12780.05053	11659.65253	17921	15867.24
10	49	0.942663	13229.33402	12122.85524	18548.87	16448.34
10	50	0.942857	13778.78384	12630.7582	19187.43	17040.63
10	51	0.943044	14267.20198	13122.76678	19836.44	17643.61
10	52	0.943223	14766.49709	13620.90169	20496.16	18257.76
10	53	0.943396	15304.25333	14151.09368	21166.35	18882.61
10	54	0.943563	15875.83738	14722.37808	21847.27	19518.61
10	55	0.943723	16431.3633	15232.26537	22538.69	20165.32
10	56	0.943878	17004.81802	15849.66668	23240.86	20823.16
10	57	0.944027	17497.28142	16335.77499	23953.56	21491.7
10	58	0.944171	18060.86609	16919.40955	24677.01	22171.37
10	59	0.94431	18709.77778	17540.45747	25411.01	22861.74
10	60	0.944444	19273.37143	18130.08679	26155.77	23563.21
10	61	0.944575	19932.04628	18742.70846	26911.1	24275.39
10	62	0.9447	20531.33008	19352.2492	27677.21	24998.66
10	63	0.944822	21147.4736	19978.4516	28453.9	25732.63
10	64	0.94494	21804.33858	20633.4589	29241.38	26477.69
10	65	0.945055	22504.55504	21334.16583	30039.45	27233.45
10	66	0.945166	23159.5771	21951.02239	30848.31	28000.28
10	67	0.945274	23866.20752	22695.34776	31667.78	28777.8
10	68	0.945378	24451.41037	23289.66683	32498.05	29566.39
10	69	0.94548	25141.53431	24010.12784	33338.94	30365.67
10	70	0.945578	25912.12806	24749.95736	34190.63	31176.01

10	71	0.945674	26592.80709	25464.74669	35052.95	31997.04
10	72	0.945767	27360.22238	26184.44881	35926.08	32829.12
10	73	0.945858	28073.89517	26914.25427	36809.85	33671.9
10	74	0.945946	28812.99728	27673.32311	37704.43	34525.7
10	75	0.946032	29583.1651	28438.59654	38609.66	35390.21
10	76	0.946115	30418.07152	29277.41446	39525.71	36265.74
10	77	0.946197	31172.70196	29999.49861	40452.4	37151.96
10	78	0.946276	32011.02452	30868.54374	41389.92	38049.21
10	79	0.946353	32690.7172	31569.73278	42338.1	38957.14
10	80	0.946429	33510.47925	32426.58793	43297.1	39876.1
10	81	0.946502	34398.60248	33277.18213	44266.77	40805.74
10	82	0.946574	35200.88589	34121.56744	45247.26	41746.39
10	83	0.946644	36074.83758	34946.06957	46238.42	42697.74
10	84	0.946712	36907.26946	35802.10186	47240.41	43660.08
10	85	0.946779	37764.59408	36679.34522	48253.08	44633.12
10	86	0.946844	38652.84094	37565.71946	49276.58	45617.16
10	87	0.946907	39623.5237	38540.33294	50310.76	46611.89
10	88	0.94697	40475.648	39363.1552	51355.76	47617.6
10	89	0.94703	41443.90056	40354.29453	52411.46	48634.01
10	90	0.94709	42226.15307	41175.7558	53477.99	49661.41
10	91	0.947148	43174.8	42162.42203	54555.21	50699.5
10	92	0.947205	44177.14863	43120.09516	55643.26	51748.57
10	93	0.947261	45103.57081	44094.59694	56742	52808.33
10	94	0.947315	46083.25561	45024.26478	57851.58	53879.07
10	95	0.947368	47036.63492	46008.01426	58971.86	54960.5
10	96	0.947421	48013.02932	47004.4344	60102.97	56052.9
10	97	0.947472	49016.52746	48004.56616	61244.79	57156
10	98	0.947522	50120.92103	49108.71775	62397.43	58270.07
10	99	0.947571	51077.34213	50047.38833	63560.79	59394.83
10	100	0.947619	52175.99698	51162.46793	64734.98	60530.55
11	24	0.936594	4645.816248	3913.530837	6962.273	6012.155
11	25	0.937391	4967.619666	4160.922488	7425.672	6401.633
11	26	0.938127	5257.5918	4426.376362	7890.929	6788.107
11	27	0.938808	5553.861063	4681.622713	8372.678	7191.932
11	28	0.939441	5907.499174	4965.800816	8856.421	7594.38
11	29	0.94003	6229.660287	5270.164765	9355.791	8013.148
11	30	0.94058	6546.57473	5547.970839	9857.182	8432.637
11	31	0.941094	6933.697466	5864.316589	10373.35	8867.735
11	32	0.941576	7297.354978	6227.107937	10894.34	9308.43
11	33	0.942029	7622.890909	6509.081137	11426.43	9761.572
11	34	0.942455	8038.370421	6855.911392	11969.28	10226.52
11	35	0.942857	8437.492754	7253.292707	12523.43	10704

11	36	0.943237	8800.444302	7591.351413	13088.54	11193.38
11	37	0.943596	9228.476961	7943.584851	13665.1	11695.34
11	38	0.943936	9663.877576	8380.0683	14252.79	12209.24
11	39	0.944259	10046.02715	8735.073223	14852.04	12735.74
11	40	0.944565	10517.74799	9148.006034	15462.55	13274.21
11	41	0.944857	10985.59708	9612.63238	16084.72	13825.27
11	42	0.945135	11403.06791	10015.67577	16718.26	14388.31
11	43	0.945399	11905.21818	10462.53452	17363.54	14963.94
11	44	0.945652	12408.20376	10961.175	18020.26	15551.55
11	45	0.945894	12856.23187	11400.92066	18688.8	16151.72
11	46	0.946125	13394.17083	11890.32298	19368.85	16763.88
11	47	0.946346	13928.05767	12414.50613	20060.77	17388.58
11	48	0.946558	14412.3799	12901.07948	20764.27	18025.26
11	49	0.946761	14987.85098	13433.58289	21479.67	18674.46
11	50	0.946957	15552.60514	13984.99101	22206.71	19335.62
11	51	0.947144	16066.93699	14503.98571	22945.69	20009.28
11	52	0.947324	16682.29744	15083.13444	23696.35	20694.91
11	53	0.947498	17280.20693	15666.47547	24458.98	21393.01
11	54	0.947665	17829.57604	16226.6055	25233.32	22103.07
11	55	0.947826	18478.8015	16839.26934	26019.67	22825.57
11	56	0.947981	19110.31204	17455.83256	26817.76	23560.02
11	57	0.948131	19695.93805	18058.29978	27627.88	24306.9
11	58	0.948276	20379.88221	18706.02723	28449.78	25065.72
11	59	0.948416	21046.54623	19358.61239	29283.71	25836.95
11	60	0.948551	21667.87395	20001.38882	30129.45	26620.12
11	61	0.948681	22383.70473	20677.80943	30987.25	27415.67
11	62	0.948808	23086.44124	21368.24259	31856.87	28223.16
11	63	0.94893	23744.70473	22052.66393	32738.57	29043.01
11	64	0.949049	24495.43665	22764.49643	33632.11	29874.79
11	65	0.949164	25233.42706	23490.65497	34537.73	30718.93
11	66	0.949275	25928.3907	24215.46293	35455.21	31574.98
11	67	0.949384	26712.78811	24960.04803	36384.79	32443.37
11	68	0.949488	27487.49832	25724.73913	37326.25	33323.68
11	69	0.94959	28218.3006	26487.31573	38279.81	34216.31
11	70	0.949689	29035.66841	27263.31301	39245.26	35120.86
11	71	0.949786	29848.89555	28069.90844	40222.82	36037.72
11	72	0.949879	30614.42784	28867.37117	41212.29	36966.48
11	73	0.94997	31465.81755	29677.17143	42213.88	37907.55
11	74	0.950059	32317.81756	30525.61083	43227.38	38860.52
11	75	0.950145	33120.12874	31362.06346	44253.01	39825.78
11	76	0.950229	34004.16918	32203.19453	45290.57	40802.94
11	77	0.950311	34892.53298	33087.4883	46340.25	41792.38

11	78	0.95039	35731.42581	33962.18696	47401.87	42793.72
11	79	0.950468	36648.978	34836.51305	48475.62	43807.32
11	80	0.950543	37576.74271	35763.08357	49561.32	44832.83
11	81	0.950617	38451.11406	36673.24835	50659.16	45870.59
11	82	0.950689	39403.2744	37583.19594	51768.95	46920.26
11	83	0.95076	40367.0011	38544.15621	52890.88	47982.17
11	84	0.950828	41276.6331	39489.41719	54024.77	49055.98
11	85	0.950895	42264.85691	40437.86403	55170.81	50142.02
11	86	0.950961	43266.01861	41436.44317	56328.81	51239.97
11	87	0.951024	44210.3526	42415.4499	57498.96	52350.14
11	88	0.951087	45234.76104	43402.46628	58681.09	53472.21
11	89	0.951148	46271.12327	44433.58355	59875.36	54606.51
11	90	0.951208	47252.85576	45452.30401	61081.62	55752.7
11	91	0.951266	48311.14867	46472.5329	62300.02	56911.11
11	92	0.951323	49386.83706	47545.2164	63530.41	58081.42
11	93	0.951379	50403.64963	48598.35713	64772.95	59263.93
11	94	0.951434	51497.98201	49656.53759	66027.49	60458.34
11	95	0.951487	52610.60366	50765.45524	67294.17	61664.96
11	96	0.95154	53662.3346	51852.56224	68572.86	62883.47
11	97	0.951591	54791.03015	52944.73571	69863.69	64114.18
11	98	0.951642	55941.76923	54092.50492	71166.53	65356.79
11	99	0.951691	57027.88879	55212.42986	72481.52	66611.58
11	100	0.951739	58194.0973	56345.45139	73808.52	67878.28
12	26	0.941538	6074.565359	5045.66608	9089.52	7738.483
12	27	0.942222	6400.213836	5315.056091	9631.597	8183.799
12	28	0.942857	6794.666667	5656.768644	10186.52	8639.651
12	29	0.943448	7154.736842	5962.189637	10749.63	9104.862
12	30	0.944	7525.39548	6277.447614	11323.61	9580.955
12	31	0.944516	7878.696721	6611.913026	11908.43	10068
12	32	0.945	8341.57672	7008.479047	12504.88	10567.26
12	33	0.945455	8749.187179	7348.55804	13112.71	11078.36
12	34	0.945882	9143.052239	7683.541043	13732.64	11602.36
12	35	0.946286	9582.550725	8067.590625	14364.3	12138.65
12	36	0.946667	10100.53286	8568.713099	15008.35	12688.19
12	37	0.947027	10539.98174	8941.660409	15664.39	13250.28
12	38	0.947368	10970.91111	9331.315866	16333	13825.81
12	39	0.947692	11449.54113	9744.289394	17013.79	14414.07
12	40	0.948	11929.56329	10187.88244	17707.31	15015.87
12	41	0.948293	12473.68107	10692.56811	18413.17	15630.53
12	42	0.948571	12941.26506	11132.32714	19131.88	16258.8
12	43	0.948837	13434.83529	11576.66657	19863.06	16900.01
12	44	0.949091	13960.40421	12116.33777	20607.12	17554.73

12	45	0.949333	14551.9588	12625.0894	21363.83	18222.57
12	46	0.949565	15120.07692	13156.96028	22133.48	18903.94
12	47	0.949787	15657.26523	13686.01449	22915.58	19597.83
12	48	0.95	16236.6193	14214.753	23710.82	20305.53
12	49	0.950204	16923.50515	14844.9114	24518.57	21025.81
12	50	0.9504	17506.3064	15405.46431	25339.52	21759.87
12	51	0.950588	18100.63696	15997.92995	26173.04	22506.56
12	52	0.950769	18734.52427	16589.8052	27019.78	23267.01
12	53	0.950943	19387.49048	17236.88012	27879.16	24040.13
12	54	0.951111	20089.65888	17885.51641	28751.79	24827
12	55	0.951273	20715.42813	18517.79211	29637.16	25626.68
12	56	0.951429	21342.84234	19112.11861	30535.75	26440
12	57	0.951579	22057.56932	19852.66624	31447.24	27266.36
12	58	0.951724	22794.45362	20510.31176	32371.99	28106.35
12	59	0.951864	23545.01994	21247.76407	33309.69	28959.39
12	60	0.952	24231.43697	21934.93335	34260.67	29826.05
12	61	0.952131	24976.073	22650.0897	35224.64	30705.77
12	62	0.952258	25817.77371	23413.11566	36201.92	31599.11
12	63	0.952381	26579.532	24179.36883	37192.22	32505.52
12	64	0.9525	27354.37139	24971.30985	38195.85	33425.53
12	65	0.952615	28142.38243	25724.59657	39212.53	34358.61
12	66	0.952727	28983.23537	26574.29554	40242.56	35305.28
12	67	0.952836	29836.03258	27361.60886	41285.67	36265.03
12	68	0.952941	30648.71481	28210.45188	42342.14	37238.36
12	69	0.953043	31407.77616	28944.16415	43411.72	38224.76
12	70	0.953143	32317.85012	29879.7523	44494.68	39224.74
12	71	0.953239	33198.99764	30683.68182	45590.76	40237.8
12	72	0.953333	34146.59907	31632.84764	46700.23	41264.41
12	73	0.953425	34979.41839	32466.46572	47822.85	42304.1
12	74	0.953514	35906.87982	33384.50376	48958.87	43357.35
12	75	0.9536	36912.36018	34306.90559	50108.05	44423.66
12	76	0.953684	37849.63135	35253.11411	51270.64	45503.52
12	77	0.953766	38808.24946	36236.02634	52446.41	46596.46
12	78	0.953846	39752.04624	37148.16433	53635.59	47702.92
12	79	0.953924	40781.09236	38184.90489	54837.97	48822.47
12	80	0.954	41779.80503	39111.75316	56053.78	49955.53
12	81	0.954074	42786.95238	40167.7497	57282.79	51101.66
12	82	0.954146	43681.88957	41049.44591	58525.24	52261.31
12	83	0.954217	44792.10505	42180.89765	59780.9	53434.04
12	84	0.954286	45816.73154	43131.30703	61050	54620.26
12	85	0.954353	46964.71598	44283.41481	62332.34	55819.56
12	86	0.954419	47946.12476	45268.67292	63628.11	57032.35

12	87	0.954483	49064.35742	46388.63607	64937.13	58258.22
12	88	0.954545	50235.32762	47475.90229	66259.59	59497.58
12	89	0.954607	51344.65254	48590.37882	67595.3	60750
12	90	0.954667	52493.81378	49768.03532	68944.46	62015.91
12	91	0.954725	53588.64365	50829.52489	70306.88	63294.89
12	92	0.954783	54805.45082	52046.78965	71682.76	64587.33
12	93	0.954839	55953.68649	53126.27317	73071.9	65892.86
12	94	0.954894	57159.72193	54385.77641	74474.5	67211.84
12	95	0.954947	58191.78571	55416.51117	75890.37	68543.9
12	96	0.955	59507.59599	56748.10226	77319.71	69889.41
12	97	0.955052	60669.82383	57835.8957	78762.32	71248
12	98	0.955102	62023.26068	59192.253	80218.4	72620.04
12	99	0.955152	63151.08883	60325.56772	81687.77	74005.15
12	100	0.9552	64460.04606	61638.55167	83170.6	75403.7
13	28	0.945767	7658.124476	6353.710663	11615.32	9794.508
13	29	0.94636	8071.731444	6664.975835	12259.37	10320.79
13	30	0.946914	8473.688396	7001.015345	12914.47	10859.11
13	31	0.947431	8923.388398	7384.334091	13581.51	11408.93
13	32	0.947917	9402.589744	7769.835013	14260.69	11970.81
13	33	0.948373	9836.569231	8155.624962	14952.72	12545.81
13	34	0.948802	10362.18944	8608.361998	15657.53	13133.94
13	35	0.949206	10813.98774	9003.457639	16375.74	13736.05
13	36	0.949588	11267.30769	9402.900478	17104.22	14346.02
13	37	0.94995	11773.72708	9857.070974	17847.38	14972.26
13	38	0.950292	12274.8441	10280.38623	18601.14	15607.03
13	39	0.950617	12863.51948	10798.58247	19369.69	16258.12
13	40	0.950926	13351.11295	11242.00663	20149.11	16918.4
13	41	0.95122	13934.01235	11742.52623	20943.35	17594.96
13	42	0.951499	14488.23262	12242.11016	21748.69	18281.39
13	43	0.951766	15063.90317	12821.16207	22568.87	18984.07
13	44	0.95202	15654.07869	13354.34656	23400.35	19697.32
13	45	0.952263	16208.70787	13842.85352	24246.68	20426.77
13	46	0.952496	16904.57058	14458.47756	25105.18	21168.44
13	47	0.952719	17463.96609	15001.75902	25977.81	21925.26
13	48	0.952932	18144.02996	15612.4397	26864.19	22696.51
13	49	0.953137	18771.29818	16186.56814	27764.8	23482.89
13	50	0.953333	19449.28283	16865.52951	28679.26	24283.71
13	51	0.953522	20136.3968	17510.30933	29608.03	25099.65
13	52	0.953704	20777.99477	18113.22873	30550.74	25930.02
13	53	0.953878	21560.57216	18803.38954	31507.81	26775.49
13	54	0.954047	22226.422	19483.25729	32478.92	27635.38
13	55	0.954209	22998.80522	20188.33059	33464.44	28510.35

13	56	0.954365	23715.97713	20865.80139	34464.06	29399.73
13	57	0.954516	24498.01157	21645.60997	35478.15	30304.16
13	58	0.954662	25280.65217	22391.60098	36506.39	31222.99
13	59	0.954802	26019.42472	23115.16208	37549.14	32156.85
13	60	0.954938	26886.66645	23880.38356	38606.09	33105.1
13	61	0.95507	27660.41767	24685.59097	39677.59	34068.34
13	62	0.955197	28525.69293	25483.67927	40763.33	35045.97
13	63	0.95532	29337.48	26269.42733	41863.65	36038.57
13	64	0.95544	30229.2338	27157.32369	42978.26	37045.55
13	65	0.955556	31110.74717	28007.27888	44107.47	38067.46
13	66	0.955668	31948.58074	28843.73354	45251	39103.75
13	67	0.955777	32899.29612	29686.23735	46409.16	40154.96
13	68	0.955882	33786.64786	30618.27411	47581.66	41220.52
13	69	0.955985	34743.38855	31505.28711	48768.82	42300.98
13	70	0.956085	35662.13226	32416.59811	49970.35	43395.79
13	71	0.956182	36656.60829	33398.74403	51186.56	44505.48
13	72	0.956276	37634.66541	34345.28454	52417.16	45629.51
13	73	0.956367	38572.47162	35290.39955	53662.44	46768.4
13	74	0.956456	39616.05076	36230.87939	54922.15	47921.62
13	75	0.956543	40607.15901	37263.34662	56196.56	49089.68
13	76	0.956628	41663.3515	38255.17047	57485.41	50272.07
13	77	0.95671	42685.64957	39277.87687	58788.97	51469.28
13	78	0.95679	43783.28784	40357.02682	60106.99	52680.82
13	79	0.956868	44863.74571	41409.70474	61439.74	53907.16
13	80	0.956944	45900.76778	42458.14011	62786.96	55147.81
13	81	0.957019	47036.73722	43494.26096	64148.91	56403.26
13	82	0.957091	48135.38697	44633.42354	65525.35	57673.01
13	83	0.957162	49290.70723	45727.30458	66916.54	58957.55
13	84	0.957231	50419.84938	46864.25509	68322.23	60256.39
13	85	0.957298	51618.31543	48035.91803	69742.67	61569.99
13	86	0.957364	52799.81332	49190.50998	71177.63	62897.9
13	87	0.957429	53933.67941	50335.45597	72627.34	64240.55
13	88	0.957492	55170.52088	51482.80965	74091.59	65597.51
13	89	0.957553	56369.53716	52712.73736	75570.59	66969.2
13	90	0.957613	57627.87237	53915.59717	77064.14	68355.19
13	91	0.957672	58862.53421	55159.72898	78572.45	69755.91
13	92	0.957729	60164.80664	56431.66979	80095.32	71170.92
13	93	0.957786	61445.05073	57682.49643	81632.95	72600.65
13	94	0.957841	62679.05923	58930.17736	83185.15	74044.66
13	95	0.957895	64014.86162	60181.42541	84752.12	75503.39
13	96	0.957948	65319.70479	61514.68657	86333.66	76976.4
13	97	0.957999	66678.06815	62817.59798	87929.98	78464.11

13	98	0.95805	68021.30769	64173.95736	89540.88	79966.11
13	99	0.9581	69426.34401	65543.96449	91166.56	81482.8
13	100	0.958148	70803.19327	66886.83713	92806.83	83013.77
14	30	0.949425	9569.387409	7860.877627	14596.51	12202.86
14	31	0.949944	10050.274	8240.550998	15345.41	12808.3
14	32	0.950431	10571.15873	8653.84801	16105.73	13423.62
14	33	0.950888	10994.20879	9054.878209	16877.86	14049.81
14	34	0.951318	11560.69083	9514.196992	17662.77	14688.65
14	35	0.951724	12039.95445	9963.919573	18460.48	15340.28
14	36	0.952107	12571.15895	10362.84797	19271.88	16006.13
14	37	0.95247	13248.18004	10968.00378	20096.75	16685.87
14	38	0.952813	13794.82095	11415.42594	20935.89	17380.7
14	39	0.953139	14361.41558	11961.5305	21788.95	18090.04
14	40	0.953448	14877.23146	12351.42257	22656.68	18814.95
14	41	0.953743	15531.77072	12899.31626	23538.64	19554.7
14	42	0.954023	16124.07057	13491.55937	24435.52	20310.29
14	43	0.95429	16725.09244	14000.90379	25346.86	21080.93
14	44	0.954545	17406.92939	14597.97358	26273.31	21867.55
14	45	0.954789	18055.75762	15183.61304	27214.41	22669.36
14	46	0.955022	18768.21821	15831.4133	28170.77	23487.26
14	47	0.955246	19482.23502	16429.37814	29141.94	24320.44
14	48	0.95546	20039.20602	17001.54575	30128.17	25169.11
14	49	0.955665	20846.40648	17695.10888	31129.66	26033.77
14	50	0.955862	21524.31457	18414.00999	32146.31	26913.93
14	51	0.956051	22230	19018.69114	33177.08	27807.55
14	52	0.956233	23082.9889	19741.38565	34223.71	28717.99
14	53	0.956409	23836.04354	20458.65238	35284.58	29642.04
14	54	0.956577	24585.46462	21155.5834	36361.36	30582.87
14	55	0.95674	25305.62516	21867.86257	37452.48	31537.49
14	56	0.956897	26201.59459	22686.18537	38559.56	32508.87
14	57	0.957048	27022.10619	23455.84166	39681.08	33494.21
14	58	0.957194	27827.46087	24264.20567	40818.58	34496.28
14	59	0.957335	28721.35043	25085.0283	41970.6	35512.52
14	60	0.957471	29584.93637	25897.79613	43138.63	36545.46
14	61	0.957603	30497.44628	26733.24458	44321.51	37593.13
14	62	0.957731	31444.55168	27608.9219	45520.17	38657.08
14	63	0.957854	32211.55771	28428.92627	46734.24	39736.62
14	64	0.957974	33213.95951	29302.70296	47964.14	40832.4
14	65	0.95809	34148.39313	30284.45685	49209.5	41943.76
14	66	0.958203	35074.27481	31160.43976	50470.72	43071.35
14	67	0.958312	36150.09452	32108.95008	51747.45	44214.51
14	68	0.958418	37137.96614	33079.58442	53040.06	45373.87

14	69	0.958521	38069.07508	33963.3752	54348.23	46548.8
14	70	0.958621	39041.64029	34969.92664	55672.32	47739.9
14	71	0.958718	40207.59068	36076.5403	57011.99	48946.56
14	72	0.958812	41219.01998	37023.64926	58367.6	50169.38
14	73	0.958904	42282.94581	38110.66365	59738.84	51407.75
14	74	0.958993	43405.68222	39169.61601	61126.03	52662.25
14	75	0.95908	44499.17066	40225.82876	62528.87	53932.29
14	76	0.959165	45626.87512	41284.0356	63947.69	55218.45
14	77	0.959248	46801.54995	42383.9846	65382.19	56520.14
14	78	0.959328	47804.0424	43471.96253	66832.69	57837.93
14	79	0.959406	49014.03003	44553.0684	68298.88	59171.24
14	80	0.959483	50218.43306	45802.40292	69781.09	60520.64
14	81	0.959557	51361.54658	46911.50126	71279.02	61885.55
14	82	0.95963	52661.56442	48081.10652	72792.97	63266.53
14	83	0.959701	53906.32208	49324.8931	74322.66	64663.02
14	84	0.95977	55024.63302	50404.38715	75868.39	66075.56
14	85	0.959838	56257.35757	51680.29978	77429.88	67503.6
14	86	0.959904	57697.21554	53056.12658	79007.42	68947.68
14	87	0.959968	58904.37572	54202.20141	80600.73	70407.26
14	88	0.960031	60218.12163	55531.66024	82210.1	71882.86
14	89	0.960093	61571.77401	56819.87607	83835.26	73373.95
14	90	0.960153	62907	58129.7779	85476.48	74881.05
14	91	0.960212	64264.91555	59429.54714	87133.52	76403.64
14	92	0.96027	65662.20141	60742.00139	88806.62	77942.22
14	93	0.960326	66918.75907	62106.48745	90495.54	79496.29
14	94	0.960382	68336.1016	63395.30831	92200.54	81066.34
14	95	0.960436	69802.64248	64887.95137	93921.38	82651.87
14	96	0.960489	71189.13463	66268.00554	95658.3	84253.38
14	97	0.96054	72696.09326	67625.80337	97411.06	85870.36
14	98	0.960591	74210.17949	69148.1046	99179.91	87503.3
14	99	0.960641	75525.47933	70439.54259	100964.6	89151.72
14	100	0.96069	77010.5341	71957.46957	102765.4	90816.09
15	32	0.952621	11646.15344	9514.504916	18020.36	14951.58
15	33	0.953079	12217.0841	9972.874584	18890.12	15658.13
15	34	0.95351	12776.81493	10415.95168	19775.41	16380.41
15	35	0.953917	13379.61256	10931.90408	20676.72	17118.98
15	36	0.954301	13922.61502	11397.48006	21594.27	17874.34
15	37	0.954664	14558.55616	11909.64499	22528.52	18646.87
15	38	0.955008	15161.43022	12422.55608	23470.09	19419.45
15	39	0.955335	15852.21732	13014.00243	24431.72	20215.04
15	40	0.955645	16413.28523	13517.41019	25401.19	21011.81
15	41	0.95594	17099.24856	14087.29581	26390.6	21831.19

15	42	0.956221	17775.51727	14672.73623	27388.24	22653.02
15	43	0.956489	18470.66745	15254.70304	28405.62	23497.04
15	44	0.956745	19154.64751	15897.9435	29431.53	24344.9
15	45	0.956989	19829.24869	16463.73302	30476.93	25214.58
15	46	0.957223	20572.67619	17117.6892	31531.11	26089.61
15	47	0.957447	21323.43584	17770.5081	32604.54	26986.13
15	48	0.957661	22061.90526	18465.70328	33686.97	27889.6
15	49	0.957867	22809.54296	19126.6032	34788.43	28814.32
15	50	0.958065	23596.99057	19825.30842	35901.33	29750.32
15	51	0.958254	24410.38218	20551.53841	37030.78	30704.51
15	52	0.958437	25215.60065	21320.38184	38176.35	31676.03
15	53	0.958612	26019.60571	22047.39908	39338.59	32665.66
15	54	0.958781	26869.08723	22815.82739	40517.08	33672.55
15	55	0.958944	27737.90642	23599.23371	41712.33	34697.48
15	56	0.959101	28611.67628	24442.67268	42923.94	35739.61
15	57	0.959253	29482.36519	25250.77752	44152.4	36799.71
15	58	0.959399	30393.09623	26084.01083	45397.3	37876.97
15	59	0.959541	31318.46211	26926.75721	46659.13	38972.11
15	60	0.959677	32259.69468	27839.67806	47937.49	40084.37
15	61	0.95981	33199.70964	28730.73638	49232.84	41214.45
15	62	0.959938	34169.49756	29623.97103	50544.79	42361.6
15	63	0.960061	35162.54453	30544.9797	51873.78	43526.51
15	64	0.960181	36168.63675	31520.76753	53219.43	44708.45
15	65	0.960298	37174.19587	32483.60596	54582.18	45908.11
15	66	0.960411	38205.78168	33441.86115	55961.64	47124.75
15	67	0.96052	39266.19599	34438.0054	57358.24	48359.05
15	68	0.960626	40339.56889	35480.94252	58771.6	49610.31
15	69	0.960729	41416.58443	36523.86801	60202.13	50879.19
15	70	0.960829	42501.6705	37530.29329	61649.48	52164.99
15	71	0.960927	43634.98109	38609.68907	63114.03	53468.37
15	72	0.961022	44775.63403	39719.23916	64595.43	54788.64
15	73	0.961114	45927.11862	40845.85482	66094.06	56126.45
15	74	0.961203	47068.00952	41905.23754	67609.58	57481.13
15	75	0.96129	48270.72528	43058.6026	69142.35	58853.31
15	76	0.961375	49476.88521	44231.00105	70692.05	60242.34
15	77	0.961458	50703.66928	45440.56996	72259.02	61648.83
15	78	0.961538	51900.95613	46554.7836	73842.94	63072.16
15	79	0.961617	53177.00934	47787.12134	75444.16	64512.92
15	80	0.961694	54444.96981	49015.36834	77062.36	65970.49
15	81	0.961768	55748.80704	50308.65834	78697.86	67445.47
15	82	0.961841	57003.80327	51480.66651	80350.38	68937.24
15	83	0.961912	58350.4796	52785.57736	82020.22	70446.39

15	84	0.961982	59686.23273	54081.7774	83707.08	71972.32
15	85	0.962049	61059.72742	55442.1199	85411.29	73515.61
15	86	0.962116	62378.51735	56683.76754	87132.54	75075.66
15	87	0.96218	63793.06782	58054.6148	88871.14	76653.04
15	88	0.962243	65194.69371	59414.41316	90626.82	78247.19
15	89	0.962305	66647.54087	60860.56387	92399.85	79858.63
15	90	0.962366	68025.4365	62161.82515	94189.97	81486.83
15	91	0.962425	69511.06409	63604.51845	95997.46	83132.31
15	92	0.962482	70979.06047	65029.10646	97822.06	84794.54
15	93	0.962539	72505.88288	66548.20649	99664.03	86474.03
15	94	0.962594	73945.84278	67915.2225	101523.1	88170.25
15	95	0.962649	75501.05432	69425.67497	103399.6	89883.72
15	96	0.962702	77031.99407	70908.15306	105293.2	91613.91
15	97	0.962754	78637.55337	72508.88068	107204.2	93361.34
15	98	0.962804	80137.20923	73936.25432	109132.4	95125.48
15	99	0.962854	81761.48596	75513.32819	111077.9	96906.84
15	100	0.962903	83363.36248	77070.1968	113040.6	98704.91
16	34	0.955437	14176.55224	11500.69803	21966.36	18095.22
16	35	0.955844	14794.35507	12020.83511	22960.09	18897.45
16	36	0.956229	15464.98239	12572.1808	23970.69	19716.71
16	37	0.956593	16159.72774	13137.72886	24995.32	20547.96
16	38	0.956938	16814.34833	13694.68231	26037.71	21397.42
16	39	0.957265	17459.28247	14228.8281	27088.95	22249.34
16	40	0.957576	18129.59335	14788.10164	28161.05	23124.71
16	41	0.957871	18800	15358.26388	29242.55	24003.73
16	42	0.958153	19552.89608	15960.60203	30344.86	24905.9
16	43	0.958421	20229.10441	16588.65223	31456.96	25812.91
16	44	0.958678	21164.91954	17353.15622	32589.7	26742.7
16	45	0.958923	21895.92978	17966.78018	33732.51	27678.61
16	46	0.959157	22701.73901	18620.62507	34895.75	28636.91
16	47	0.959381	23468.99731	19348.86076	36069.25	29602.66
16	48	0.959596	24146.55526	19972.46378	37262.95	30590.49
16	49	0.959802	24988.52448	20696.01887	38467.09	31587.16
16	50	0.96	25888.88258	21543.63376	39691.18	32605.66
16	51	0.96019	26819.26485	22366.32408	40927.96	33636.86
16	52	0.960373	27630.43083	23059.75636	42182.34	34687.14
16	53	0.960549	28556.31786	23858.92625	43453.84	35755.59
16	54	0.960718	29493.30958	24715.51208	44743.1	36843.11
16	55	0.960882	30412.04587	25542.83811	46049.62	37948.81
16	56	0.961039	31402.69369	26447.30582	47374.03	39073.57
16	57	0.961191	32166.79425	27202.99109	48715.84	40216.51
16	58	0.961338	33183.5337	28157.74846	50075.63	41378.48

16	59	0.961479	34123.63782	28957.60894	51452.96	42558.61
16	60	0.961616	35122.76891	29950.69204	52848.36	43757.76
16	61	0.961749	36331.63223	30969.18845	54261.39	44975.05
16	62	0.961877	37328.04065	31891.47371	55692.58	46211.32
16	63	0.962001	38368.614	32815.99125	57141.48	47465.73
16	64	0.962121	39453.20374	33860.98307	58608.62	48739.07
16	65	0.962238	40371.86047	34802.00838	60093.55	50030.54
16	66	0.962351	41516.33874	35875.57186	61596.77	51340.91
16	67	0.96246	42720.53853	36991.46992	63117.87	52669.38
16	68	0.962567	43939.6963	38103.98707	64657.3	54016.72
16	69	0.96267	45020.94526	39137.07672	66214.68	55382.15
16	70	0.962771	46212.76079	40209.73745	67790.43	56766.41
16	71	0.962868	47499.9406	41448.90735	69384.18	58168.74
16	72	0.962963	48693.03759	42575.50492	70996.36	59589.86
16	73	0.963055	49938.04569	43705.66953	72626.59	61029.04
16	74	0.963145	51030.97959	44877.86239	74275.27	62486.99
16	75	0.963232	52349.02936	46129.25955	75942.05	63962.97
16	76	0.963317	53563.38825	47263.04304	77627.32	65457.69
16	77	0.9634	54872.79167	48558.72244	79330.73	66970.43
16	78	0.963481	56391.16694	49879.46604	81052.66	68501.88
16	79	0.96356	57677.7914	51117.26735	82792.75	70051.33
16	80	0.963636	59033.54403	52410.40345	84551.4	71619.46
16	81	0.963711	60397.47516	53717.35706	86328.26	73205.59
16	82	0.963784	61623.17331	55010.62878	88123.68	74810.36
16	83	0.963855	63071.95606	56389.90284	89937.34	76433.12
16	84	0.963925	64573.79117	57780.04835	91769.6	78074.5
16	85	0.963993	66104.82027	59208.81643	93620.12	79733.86
16	86	0.964059	67496.96637	60594.82943	95489.25	81411.81
16	87	0.964124	68960.68569	61943.22865	97376.66	83107.73
16	88	0.964187	70595.89	63520.99841	99282.71	84822.22
16	89	0.964249	72105.84816	64991.37876	101207.1	86554.66
16	90	0.96431	73609.29958	66369.78302	103150.1	88305.66
16	91	0.964369	75031.53384	67895.67099	105111.4	90074.6
16	92	0.964427	76661.85519	69453.19925	107091.4	91862.08
16	93	0.964484	78173.39865	70917.10199	109089.8	93667.49
16	94	0.964539	79804.35561	72528.34858	111106.8	95491.42
16	95	0.964593	81634.38624	74146.87409	113142.2	97333.27
16	96	0.964646	83238.94764	75728.46722	115196.3	99193.62
16	97	0.964699	84896.57383	77323.46941	117268.7	101071.9
16	98	0.96475	86552.725	78914.27688	119359.9	102968.6
16	99	0.9648	88094.46193	80546.7704	121469.4	104883.3
16	100	0.964848	89860.22236	82243.13896	123597.7	106816.4

17	36	0.957937	16956.15162	13678.01115	26434.39	21641.92
17	37	0.958301	17564.84206	14187.47468	27560.42	22546.37
17	38	0.958647	18416.93882	14886.97793	28704.85	23469.77
17	39	0.958974	19211.07181	15498.05893	29868.16	24412.54
17	40	0.959286	19831.63589	16070.44481	31040.11	25356.57
17	41	0.959582	20632.77342	16679.92232	32234.83	26326.39
17	42	0.959864	21432.49539	17381.53521	33438.84	27298.6
17	43	0.960133	22266.43391	18133.65298	34665.61	28296.26
17	44	0.96039	22986.0764	18685.69417	35902.14	29297.48
17	45	0.960635	23833.22736	19384.21314	37161.27	30323.7
17	46	0.96087	24579.25598	20061.61612	38430.51	31354.73
17	47	0.961094	25699.38204	21047.34263	39722.12	32410.3
17	48	0.96131	26496.89721	21670.96493	41024.08	33472.02
17	49	0.961516	27247.64888	22376.55795	42348.14	34557.85
17	50	0.961714	28177.79857	23139.08497	43682.74	35651.24
17	51	0.961905	29143.62434	24016.79579	45039.16	36768.39
17	52	0.962088	30288.27356	25108.79395	46406.27	37894.6
17	53	0.962264	30994.11485	25670.83768	47794.94	39044.28
17	54	0.962434	31969.8829	26477.79323	49196.86	40207.29
17	55	0.962597	32833.37561	27306.64118	50617.62	41390.68
17	56	0.962755	34077.84473	28343.80091	52056.72	42593.53
17	57	0.962907	35081.04373	29257.82635	53514.8	43816.76
17	58	0.963054	36001.21586	30187.05811	54991.38	45059.45
17	59	0.963196	37044.43992	31062.26436	56487.07	46322.49
17	60	0.963333	38187.2264	32123.96604	58001.39	47604.99
17	61	0.963466	39411.05348	33197.67653	59534.94	48907.81
17	62	0.963594	40369.27642	34164.50025	61087.25	50230.07
17	63	0.963719	41456.36	35077.21097	62658.87	51572.62
17	64	0.963839	42476.04308	36091.49209	64249.37	52934.6
17	65	0.963956	43884.99544	37266.48047	65859.26	54316.82
17	66	0.964069	45066.49798	38383.40045	67488.12	55718.46
17	67	0.964179	46168.47811	39524.51554	69136.45	57140.31
17	68	0.964286	47352.85708	40567.19693	70803.83	58581.54
17	69	0.964389	48696.96909	41828.8281	72490.75	60042.96
17	70	0.96449	50048.92383	43007.44367	74196.8	61523.73
17	71	0.964588	51205.95786	44221.21824	75922.44	63024.65
17	72	0.964683	52417.14068	45279.6734	77667.27	64544.91
17	73	0.964775	53603.83489	46482.77345	79431.75	66085.27
17	74	0.964865	55185.28491	47816.25291	81215.48	67644.96
17	75	0.964952	56540.82235	49116.23857	83018.9	69224.71
17	76	0.965038	57827.66848	50453.8036	84841.62	70823.77
17	77	0.965121	59159.52749	51662.5852	86684.09	72442.85

17	78	0.965201	60672.73359	53067.65467	88545.9	74081.22
17	79	0.96528	62198.47171	54435.63078	90427.48	75739.59
17	80	0.965357	63541.78727	55849.30068	92328.46	77417.23
17	81	0.965432	64906.95396	57096.88286	94249.25	79114.83
17	82	0.965505	66271.59257	58498.3549	96189.47	80831.69
17	83	0.965577	68015.76649	59973.94895	98149.52	82568.48
17	84	0.965646	69554.19126	61465.24127	100129	84324.5
17	85	0.965714	71026.60599	62991.2338	102128.4	86100.43
17	86	0.965781	72515.84348	64374.7719	104147.3	87895.58
17	87	0.965846	74193.12377	65924.19474	106186.1	89710.61
17	88	0.965909	75899.21916	67480.35488	108244.4	91544.84
17	89	0.965971	77430.06082	69086.20787	110322.6	93398.94
17	90	0.966032	78951.58626	70514.13135	112420.4	95272.22
17	91	0.966091	80494.01592	72101.49253	114538.1	97165.33
17	92	0.966149	82403.17743	73728.23269	116675.4	99077.62
17	93	0.966206	84126.17202	75406.56351	118832.6	101009.7
17	94	0.966261	85783.6178	77114.5775	121009.5	102961
17	95	0.966316	87428.1329	78662.15218	123206.3	104932
17	96	0.966369	89274.61195	80369.82656	125422.7	106922.3
17	97	0.966421	91167.49985	82119.02765	127659.1	108932.2
17	98	0.966472	92878.26275	83896.70888	129915.2	110961.4
17	99	0.966522	94568.32816	85518.47213	132191.2	113010.2
17	100	0.966571	96285.01123	87278.63194	134486.9	115078.2
18	38	0.960171	20074.08148	16129.5132	31468.74	25617.16
18	39	0.960499	20805.46753	16742.93161	32735.08	26630.48
18	40	0.960811	21601.27848	17373.09772	34021.95	27664.59
18	41	0.961107	22551.16735	18206.94313	35317.59	28699.74
18	42	0.96139	23264.47122	18786.60942	36638.13	29762.62
18	43	0.961659	24111.76601	19506.14826	37968.08	30827.68
18	44	0.961916	24981.62324	20229.04017	39322.79	31919.95
18	45	0.962162	25925.06742	20936.46187	40687.35	33015.62
18	46	0.962397	26850.05372	21732.67251	42076.44	34137.91
18	47	0.962622	27773.51493	22562.81826	43475.67	35264.92
18	48	0.962838	28689.18363	23294.85631	44899.1	36417.98
18	49	0.963045	29676.81443	24076.51482	46332.87	37577.15
18	50	0.963243	30575.62851	24887.96072	47790.49	38761.85
18	51	0.963434	31555.25633	25724.29914	49258.59	39954.15
18	52	0.963617	32460.8781	26583.98588	50750.14	41171.56
18	53	0.963794	33567.53122	27530.94677	52252.29	42398.16
18	54	0.963964	34468.63344	28330.58038	53777.52	43649.52
18	55	0.964128	35671.57187	29316.24136	55316.24	44914.65
18	56	0.964286	36715.72673	30253.53881	56874.83	46201.2

18	57	0.964438	37865.97837	31224.92948	58452.77	47508.17
18	58	0.964585	38955.33333	32260.51256	60050.75	48836.6
18	59	0.964727	40089.88224	33244.54191	61668.27	50185.5
18	60	0.964865	41217.31839	34247.19847	63306	51555.88
18	61	0.964998	42329.46924	35328.33637	64963.42	52946.76
18	62	0.965126	43497.41102	36337.48362	66641.18	54359.12
18	63	0.965251	44677.17422	37415.81677	68338.78	55792.01
18	64	0.965372	45929.89501	38548.57621	70056.83	57246.36
18	65	0.965489	47192.40138	39669.7389	71794.86	58721.25
18	66	0.965602	48443.83376	40836.37857	73553.44	60217.58
18	67	0.965712	49771.82373	42057.22318	75332.1	61734.46
18	68	0.965819	51078.2963	43172.68589	77131.4	63272.76
18	69	0.965922	52394.25142	44515.01604	78950.89	64831.59
18	70	0.966023	53634.27818	45620.37721	80791.1	66411.82
18	71	0.966121	54916.10481	46907.35138	82651.58	68012.58
18	72	0.966216	56375.12432	48185.98086	84532.86	69634.71
18	73	0.966309	57682.31494	49496.72271	86434.48	71277.36
18	74	0.966399	59206.72562	50800.46698	88356.97	72941.35
18	75	0.966486	60618.79567	52140.05459	90299.87	74625.84
18	76	0.966572	62087.98381	53439.10722	92263.68	76331.64
18	77	0.966655	63544.50545	54842.56629	94247.98	78057.94
18	78	0.966736	65076.50681	56269.97098	96253.25	79805.52
18	79	0.966815	66561.7799	57662.88806	98279.05	81573.57
18	80	0.966892	68088.19217	59186.3442	100325.9	83362.88
18	81	0.966967	69559.1677	60519.78702	102393.3	85172.66
18	82	0.96704	71121.89775	62003.48396	104481.7	87003.65
18	83	0.967112	72774.78721	63566.13103	106590.8	88855.1
18	84	0.967181	74390.78576	65047.7893	108721	90727.74
18	85	0.96725	76047.16897	66641.99189	110871.9	92620.81
18	86	0.967316	77776.69461	68271.72508	113043.9	94535.06
18	87	0.967381	79448.03982	69806.7309	115236.6	96469.73
18	88	0.967445	81164.47048	71492.10065	117450.4	98425.54
18	89	0.967507	82780.45323	73059.85506	119685	100401.8
18	90	0.967568	84471.37058	74773.35099	121940.8	102399.1
18	91	0.967627	86280.60528	76391.15488	124217.4	104416.8
18	92	0.967685	87989.87735	78104.85025	126515.2	106455.6
18	93	0.967742	89918.9045	79867.74842	128833.8	108514.9
18	94	0.967798	91700.00654	81568.65504	131173.6	110595.1
18	95	0.967852	93554.42269	83289.9899	133534.3	112695.8
18	96	0.967905	95391.40256	85069.61479	135916.2	114817.5
18	97	0.967958	97310.29188	86878.5312	138319	116959.5
18	98	0.968009	99183.73789	88678.30629	140743	119122.6

18	99	0.968059	101127.3976	90609.50984	143188	121306.1
18	100	0.968108	102959.2831	92348.19284	145654.2	123510.5
19	40	0.962179	23449.27049	18864.29261	37125.62	30090.94
19	41	0.962477	24423.21702	19602.60769	38543.65	31221.59
19	42	0.962759	25224.35954	20243.3631	39989.65	32383.52
19	43	0.963029	26170.46502	21034.68067	41453.69	33560.22
19	44	0.963287	27113.82033	21790.26523	42924.3	34733.34
19	45	0.963533	28046.40982	22638.77733	44423.7	35938.24
19	46	0.963768	29029.57953	23351.64108	45930.21	37140.85
19	47	0.963993	29943.93492	24118.08612	47465.07	38374.34
19	48	0.964209	31024.32742	25055.95255	49007.41	39607.04
19	49	0.964417	32065.06077	25940.68658	50577.61	40869.76
19	50	0.964615	33010.47156	26776.97291	52155.56	42133.39
19	51	0.964806	34116.15373	27628.47658	53760.83	43426.28
19	52	0.96499	35137.37302	28500.69954	55374.04	44721.92
19	53	0.965167	36268.84261	29501.18099	57014.02	46046.17
19	54	0.965337	37318.2179	30384.97034	58662.09	47375.17
19	55	0.965501	38480.46692	31519.04216	60336.39	48732.24
19	56	0.965659	39592.78189	32343.29108	62018.9	50096.16
19	57	0.965812	40710.79506	33312.32882	63727.13	51487.75
19	58	0.965959	41956.97712	34441.20304	65447.68	52892.39
19	59	0.966102	43102.40171	35444.42086	67189.38	54320.13
19	60	0.966239	44348.13401	36636.14083	68951.7	55769.94
19	61	0.966372	45563.01392	37611.45696	70735.34	57242.82
19	62	0.966501	46780.48224	38695.58956	72539.79	58737.77
19	63	0.966626	48119.96	39911.28103	74365.71	60255.77
19	64	0.966747	49361.77	41029.38051	76212.59	61795.81
19	65	0.966864	50745.541	42355.83009	78081.09	63358.86
19	66	0.966977	52012.82001	43403.01001	79970.69	64943.93
19	67	0.967088	53325.62129	44593.8175	81882.02	66551.96
19	68	0.967195	54772.52749	45917.70233	83814.58	68181.99
19	69	0.967298	56110.52516	47143.32522	85768.97	69834.94
19	70	0.967399	57599.97312	48553.54424	87744.71	71509.86
19	71	0.967497	58971.54573	49753.18104	89742.37	73207.64
19	72	0.967593	60379.0979	51044.89116	91761.47	74927.37
19	73	0.967685	61954.09038	52511.41508	93802.57	76669.91
19	74	0.967775	63376.33477	53821.29803	95865.21	78434.37
19	75	0.967863	64967.98905	55316.20797	97949.93	80221.6
19	76	0.967949	66440.72813	56644.67234	100056.3	82030.71
19	77	0.968032	67962.44754	58071.84748	102184.7	83862.55
19	78	0.968113	69646.40917	59641.5297	104334.9	85716.24
19	79	0.968192	71161.01542	61048.52193	106507.2	87592.61

19	80	0.968269	72857.64019	62637.06223	108701.3	89490.82
19	81	0.968344	74442.53122	64105.29251	110917.7	91411.65
19	82	0.968418	76068.60155	65644.82691	113155.9	93354.29
19	83	0.968489	77868.36587	67327.55717	115416.3	95319.53
19	84	0.968559	79479.84021	68838.88293	117698.7	97306.54
19	85	0.968627	81275.20118	70511.62971	120003.3	99316.11
19	86	0.968694	82979.21484	72123.33701	122329.9	101347.4
19	87	0.968759	84708.65926	73771.73476	124678.9	103401.3
19	88	0.968823	86628.69624	75574.43372	127049.8	105476.8
19	89	0.968885	88330.96075	77176.37437	129443.2	107574.9
19	90	0.968946	90232.86945	78949.49703	131858.5	109694.6
19	91	0.969005	92065.02035	80714.85996	134296.3	111836.8
19	92	0.969064	93892.72332	82460.46039	136756.1	114000.7
19	93	0.96912	95925.96956	84367.55179	139238.4	116187.1
19	94	0.969176	97725.23755	86072.02842	141742.8	118395
19	95	0.969231	99733.132	87945.00202	144269.6	120625.4
19	96	0.969284	101688.191	89848.58085	146818.6	122877.4
19	97	0.969337	103618.5653	91696.5511	149390.1	125151.8
19	98	0.969388	105772.505	93723.35281	151983.7	127447.8
19	99	0.969438	107669.9768	95531.15811	154599.9	129766.2
19	100	0.969487	109781.319	97498.48374	157238.2	132106.1
20	42	0.963995	27319.2253	21844.49994	43439.71	34997.96
20	43	0.964265	28335.51176	22619.99412	44998.98	36218.14
20	44	0.964523	29345.98506	23493.20574	46584.59	37463.53
20	45	0.96477	30295.61685	24256.5903	48181.8	38717.44
20	46	0.965005	31452.6989	25153.3042	49805	39996.17
20	47	0.965231	32453.76452	26033.91224	51439.91	41284.65
20	48	0.965447	33568.84	26885.90088	53100.36	42597.61
20	49	0.965655	34709.20412	27892.91542	54772.54	43921.62
20	50	0.965854	35847.06162	28817.08743	56469.74	45269.84
20	51	0.966045	36886.63663	29646.83333	58178.62	46630.52
20	52	0.966229	38002.81942	30575.2322	59911.96	48015.28
20	53	0.966406	39177.52857	31649.17872	61659.73	49416
20	54	0.966576	40368.27664	32660.43407	63428.5	50838.6
20	55	0.966741	41564.53853	33567.97705	65217.81	52282.29
20	56	0.966899	42587.86306	34516.42667	67028.56	53748.42
20	57	0.967052	43748.98319	35557.90652	68860.22	55236.08
20	58	0.967199	45260.07565	36731.27842	70713.66	56746.55
20	59	0.967342	46362.25812	37759.95987	72588.33	58278.86
20	60	0.96748	47998.13109	39181.54936	74485.04	59834.23
20	61	0.967613	49079.06446	40144.25198	76403.22	61411.65
20	62	0.967742	50381.57805	41296.81512	78343.67	63012.3

20	63	0.967867	51654.844	42484.97741	80305.81	64635.18
20	64	0.967988	52999.89843	43590.72621	82290.41	66281.42
20	65	0.968105	54467.67597	44899.98004	84296.9	67950.03
20	66	0.968219	56028.61527	46312.14612	86326	69642.09
20	67	0.968329	57316.5	47433.55979	88377.18	71356.62
20	68	0.968436	58767.03185	48702.68119	90451.11	73094.67
20	69	0.96854	60480.88978	50285.04155	92547.27	74855.29
20	70	0.968641	61880.71295	51527.32707	94666.32	76639.47
20	71	0.968739	63188.82128	52697.98882	96807.73	78446.27
20	72	0.968835	64783.32797	54116.17691	98972.14	80276.68
20	73	0.968927	66281.47379	55469.40919	101159	82129.76
20	74	0.969018	67935.74354	57047.34505	103369	84006.47
20	75	0.969106	69590.63826	58562.90208	105601.6	85905.87
20	76	0.969191	71242.51126	60004.87585	107857.4	87828.92
20	77	0.969275	72617.32288	61412.92509	110135.9	89774.7
20	78	0.969356	74211.25677	62929.96694	112437.7	91744.13
20	79	0.969435	76179.97962	64561.31453	114762.2	93736.3
20	80	0.969512	77679.50314	66031.63909	117110.1	95752.12
20	81	0.969587	79727.83168	67786.98494	119480.9	97790.69
20	82	0.969661	81369.72945	69443.63783	121875	99852.91
20	83	0.969733	83134.30303	71078.03576	124292.1	101937.9
20	84	0.969803	84802.20599	72644.55855	126732.7	104046.5
20	85	0.969871	86648.58047	74407.41748	129196.3	106177.9
20	86	0.969938	88511.90819	76067.58802	131683.4	108332.9
20	87	0.970003	90529.11214	77874.50829	134193.6	110510.7
20	88	0.970067	92359.14971	79693.32437	136727.4	112712.1
20	89	0.970129	94194.39209	81356.22821	139284.2	114936.3
20	90	0.97019	96386.01229	83302.41792	141864.7	117184
20	91	0.970249	98293.67845	85148.0853	144468.4	119454.5
20	92	0.970308	100083.165	86962.51193	147095.7	121748.6
20	93	0.970365	102125.2411	88823.48104	149746.3	124065.5
20	94	0.97042	104053.4332	90641.04626	152420.5	126405.9
20	95	0.970475	106218.845	92703.17298	155118	128769.1
20	96	0.970528	108339.0089	94670.89041	157839.2	131155.9
20	97	0.970581	110454.1332	96615.39227	160583.7	133565.3
20	98	0.970632	112329.9723	98597.23452	163352	135998.3
20	99	0.970682	114423.4523	100656.1766	166143.7	138454.1
20	100	0.970732	116845.7839	102707.8689	168959.1	140933.4

APPENDIX B:
CONTRIBUTION 2 EXPERIMENTAL RESULTS

No.	k	h	s	M	Expected Search Time	Standard Deviation
1	4	10	1	2	356.0933	225.7632
2	7	50	1	5	19755.97	6637.184
3	3	8	4	1	2157.344	2061.314
4	8	50	4	4	465021.4	177864.4
5	2	6	2	6	382.5876	129.6914
6	6	50	1	9	19315.34	4170.292
7	5	12	3	8	12585.3	2574.178
8	9	50	3	9	359825.1	74310.78
9	4	10	2	4	2323.305	863.6741
10	6	50	1	5	16489	5727.711
11	3	8	3	2	1807.196	1130.376
12	6	50	3	1	88073.25	85000.47
13	5	12	1	9	1137.632	210.6099
14	10	50	1	9	34608.32	6751.602
15	2	6	3	7	935.4315	252.6608
16	7	50	3	10	280467.3	55106.17
17	2	6	2	3	287.0621	157.4899
18	7	50	1	1	8622.408	8560.331
19	3	8	3	5	2661.003	866.7255
20	10	50	4	3	533778.1	239939.6
21	2	6	2	9	432.3796	109.0173
22	8	50	1	7	25427.19	6369.172

23	4	10	3	6	6262.348	1667.501
24	9	50	3	8	351024.4	79420.46
25	3	8	2	4	1042.675	408.5275
26	9	50	1	2	18235.86	11101.82
27	5	12	3	1	5219.528	4698.394
28	7	50	4	4	401284.9	156341.4
29	5	12	1	9	1137.632	210.6099
30	9	50	2	7	142291	35659.43
31	4	10	4	6	11394.12	3025.656
32	6	50	4	9	425494.7	92285.65
33	4	10	2	1	1118.645	1064.895
34	10	50	1	5	30022.64	9328.617
35	2	6	4	1	672.3636	676.0905
36	10	50	4	2	444278.6	260189.7
37	4	10	1	6	555.1513	145.3047
38	9	50	1	7	29059.35	7133.794
39	2	6	3	10	1020.294	204.1091
40	9	50	4	9	658756.6	135834.7
41	3	8	1	3	186.8592	100.1725
42	7	50	2	1	43905.76	41843.3
43	5	12	4	1	9514.135	8519.574
44	10	50	4	1	296731.9	265134.1
45	2	6	2	9	432.3796	109.0173

46	6	50	1	7	18204.39	4842.797
47	4	10	3	6	6262.348	1667.501
48	7	50	4	10	514128.7	100777.3
49	2	6	1	3	59.45649	39.35569
50	10	50	2	5	146132.8	46079.59
51	3	8	3	4	2466.94	949.4665
52	6	50	4	1	162141.1	155616
53	3	8	1	8	268.6401	63.29729
54	10	50	2	9	168837.4	34263.51
55	5	12	4	8	22852.05	4666.681
56	8	50	4	8	566733.3	130097.7
57	3	8	2	5	1127.057	372.034
58	7	50	1	1	8622.408	8560.33
59	4	10	4	2	7508.888	4448.49
60	10	50	4	5	638416.2	201199
61	2	6	1	7	90.18959	30.6437
62	7	50	2	9	114442.6	24500.2
63	3	8	4	9	5616.087	1126.67
64	7	50	4	10	514128.7	100777
65	2	6	2	4	327.7187	147.8576
66	9	50	2	1	59450.07	54475.79
67	4	10	4	5	10833.88	3315.345
68	6	50	4	4	337763.4	134155.4

69	5	12	1	7	1080.575	243.8161
70	8	50	1	9	26885.97	5476.432
71	2	6	3	10	1020.294	204.1091
72	9	50	4	8	642673.7	145185.6
73	5	12	1	3	827.9878	368.1163
74	8	50	2	4	105754	40658.05
75	3	8	4	4	4491.784	1707.037
76	7	50	3	2	161124.8	99940.29
77	3	8	1	6	247.4705	75.45106
78	8	50	1	10	27436.97	5112.862
79	4	10	4	8	12184.32	2570.382
80	7	50	4	6	456507.8	132642