REPORT DOCUMENTATION PAGE

Form Approved
OME No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*: -06-2018

2. REPORT TYPE: Final Technical Report

3. DATES COVERED *(From - To)*: From 7/1/2016 to 12/31/2017

4. TITLE AND SUBTITLE
The Design of Responsive Sea-Based Logistic Delivery Systems

5a. CONTRACT NUMBER

5b. GRANT NUMBER
N000141612870

5c. PROGRAM ELEMENT NUMBER

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

6. AUTHOR(S)
Pazour, Jennifer A.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Rensselaer Polytechnic Institute
110 8th street
Troy, NY 12180-3522

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research
495 Summer Street Room 627
Boston MA 02210-2109

10. SPONSOR/MONITOR'S ACRONYM(S)
ONR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for Public Release; distribution is unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This project contributed fundamental insights to the field of military logistics about the emerging practice of sea-basing. The research can help transform vessels into floating distribution centers, responsible for fulfilling emergent and personalized resupply orders from troops on shore. This is the first to explicitly model internal cargo flow processes in a seabased logistics environment. Compared to traditional logistics, sea-based logistics have limited and highly constrained storage space. In very dense storage environments with selective offloading, this is the first to mathematically model and analyze uncertainty propagation, define metrics that quantify and characterize uncertainty in unit location, create analytical expressions to estimate retrieval and searching efforts, and evaluate asset tracking technology. New methodologies to analyze the impact of activity profile based cost functions in a newsvendor modeling framework are developed. A value model assessed asset tracking technologies; internal positioning systems are the preferred technology; RFID is not found adequate for the requirements of selective offloading in dense storage systems.

15. SUBJECT TERMS
sea-basing; dense storage; asset tracking technologies

16. SECURITY CLASSIFICATION OF: a. REPORT	b. ABSTRACT	c. THIS PAGE	17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
UU	UU	UU	UU	177	Jennifer Pazour

19b. TELEPHONE NUMBER *(Include area code)*: 1-518-276-6486

Final Technical Report for
Young Investigator Program: The Design of Responsive Sea-Based Logistic Delivery Systems
Award N000141612870

Submitted by
PI: Dr. Jennifer A Pazour
June 2018

This report is our final deliverable from the research conducted at Rensselaer Polytechnic Institute for *Young Investigator Program: The Design of Responsive Sea-Based Lcgistic Delivery Systems, Award Number* N000141612870. It describes the efforts under the Period of Performance of 7/1/2016 to 12/31/2017.

to Technical Representative:
ONR EXP WAREFARE & COMBATING TERORISM DE.
875 N. Randolph Street
Arlington VA 22203-1995
Billy Short, Program Director (Billy.Short@Navy.Mil) Code 30
(703) 696-0842

Approved for Public Release; Distribution is Unlimited

PI Contact Information:

Jennifer Pazour, Ph.D.
Assistant Professor of Industrial and Systems Engineering
Rensselaer Polytechnic Institute (RPI)
110 8th Street, Troy, NY 12180

Email: Pazouj@rpi.edu
Office Phone: 1 (518) 276-6486
Webpage: https://jenpazour.wordpress.com/

Abstract and Technical Summary

In the field of military logistics, this project contributed fundamental insights to the emerging practice of sea-basing. Our research can help transform vessels into floating distribution centers, responsible for fulfilling emergent and personalized resupply orders from troops on shore and can inform the direction of future technology and process innovations. This work represents the first to explicitly model internal cargo flow processes in a seabased logistics environment. Compared to traditional logistics, sea-based logistics have limited and highly constrained storage space. In very dense storage environments with selective offloading, our research team is the first to mathematically model and analyze uncertainty propagation, define metrics that quantify and characterize uncertainty in unit location, create analytical expressions to estimate retrieval and searching efforts, and evaluate asset tracking technology. By modeling the unique features of a sea-based logistics environment on proactive and reactive fulfilment decisions, we developed new methodologies to analyze the impact of activity profile based cost functions in a newsvendor modeling framework. A value model, using insights from subject matter experts, is developed to assess asset tracking technologies; internal positioning systems are the preferred asset tracking technology; RFID is not found to be adequate, given the requirements of selective offloading in dense storage systems.

List of Students Supported and Degrees Conferred

The following students were supported via this research grant. Their degrees conferred, as well as their periods of research support are provided. The students are listed in order, based on their first time of involvement.

1. Shahab Mofidi, supported as a graduate student researcher from August 2016 to December 2017
   a. Expected graduation with Ph.D. in Decision Sciences and Engineering Systems in December 2018.
2. Kaan Unnu, supported as a graduate student researcher from August 2016 to December 2017.
   a. Expected graduation with a Ph.D. in Decision Sciences and Engineering Systems in December 2019.
3. Joan Climes, supported as an undergraduate student researcher from September 2016 to May 2017.
   a. Graduated with a BS in Industrial and Management Engineering in May 2017, and a MS in Industrial and Management Engineering in December 2017. Thesis: *"Analytical Models for Retrieving Items in Dense Storage Systems and Optimizing the Location of an Open Square,"* Defended Fall 2017. Awarded the 2018 Del and Ruth Karger Outstanding Master's Thesis Award from RPI's Industrial and Systems Engineering Department.
4. Ian Shin, supported as an undergraduate student researcher in Summer 2016.
   a. Graduated with a BS in Industrial and Management Engineering in 2017.

List of Articles Produced

The following articles have been produced through this project, and constitute the body of this report. These articles follow as separate chapters.

Chapter 1: Page 5 – 18

1. Pazour, Jennifer A. and Shin, Ian, 2016, "Logistics Models to Support Order-Fulfillment from the Sea" *Progress in Material Handling Research: 2016*, Material Handling Institute, Charlotte, NC; 1-17.

Chapter 2: Page 19-43

2. Reilly, Patrick, Pazour, Jennifer A., and Schneider, Kellie, 2017, "Propagation of Unit Location Uncertainty in Dense Storage Seabasing Environments," *International Journal of Production Research*, 55(18), 5435-5449.

Chapter 3: Page 44-65

3. Scala, Natalie, and Pazour, Jennifer A., 2016, "A Value Model for Asset Tracking Technology to Support Naval Sea-based Resupply," *Engineering Management Journal* (Special Issue on Engineering Management in the Military), 28.2, 120-130.

Chapter 4: Page 66-101

4. Mofidi, Shahab, Pazour, Jennifer A., and Roy, Debit, 2018, "Proactive vs. Reactive Resource Allocation for Sea-based Logistics," *Transportation Research Part E: Logistics and Transportation Review*, 114, 66-84.

Chapter 5: Page 102-139

5. Climes, Joan, and Pazour, Jennifer A., "Analytical Models for Retrieving and Repositioning Items in Dense Storage Systems and Optimizing the Location of an Open Square," working paper.

Chapter 6: Page 140-167

6. Awwad, Mohamed, and Pazour, Jennifer A., 2018, "Search plan for a single item in an inverted T k-deep storage system," *Military Operations Research*, 23, 1-18.

Chapter 7: Page 168-177

7. Awwad, Mohamed, and Pazour, Jennifer A., 2017, "Batch Order Picking in a Dense Storage System with Item Location Uncertainty," *Proceedings of the 2017 Industrial and Systems Engineering Research Conference, 1619-1624.*

Chapter 1: Logistics Models to Support Order-Fulfillment from the Sea

This chapter has been published in the following peer-reviewed journal:

Pazour, Jennifer A. and Shin, Ian, 2016, "Logistics Models to Support Order-Fulfillment from the Sea" *Progress in Material Handling Research: 2016,* Material Handling Institute, Charlotte, NC; 1-17.

LOGISTICS MODELS TO SUPPORT ORDER-FULFILLMENT FROM THE SEA

Abstract

Sea based logistics use maritime platforms to transfer cargo stored on vessels and delivers them ashore. This chapter describes the motivations and logistical requirements of seabasing. The sea base's organizational structure, its material handling environment, and the internal cargo flow processes of the T-AKE vessel are described. Three seabasing distribution network scenarios -- *Iron Mountain*, *Skin-to-Skin Replenishment*, and *Tailored Resupply Packages* -- are described and mapped to warehousing and distribution networks, characteristics, and decision problems. Finally, related literature is reviewed and open areas for logistics research to support order fulfillment from the sea are identified.

1. Introduction to Seabasing

An important focus of military maritime logistics is the ability to support the needs of globally distributed and networked expeditionary forces that allows for increased effectiveness of resources, strategic agility, and responsiveness [12]. To meet this requirement, seabasing -- a concept using maritime platforms to transfer cargo stored on vessels and delivers them ashore -- can be used to forward deploy a set of distributed resources. These distributed resources are leveraged to provide on-time delivery of mission, operational and logistical material, equipment and personnel [12]. Strategically, seabasing provides flexibility in the ability to conduct a wide range of missions, including humanitarian aid distribution, crisis prevention, combat operations, and operational and tactical sustainment of military forces on the ground [4].

The primary motivations for seabasing operations is to increase mission flexibility, and to reduce the lead time of troop deployment and supply delivery. When a military operates without a seabase, cargo requested to combat zones or to areas in need of humanitarian aid, must first be picked from a warehouse on land, transported to the closest port, and finally loaded onto a vessel where the cargo is then delivered to its destination. By having a pre-established and stocked base in the sea, seabasing is a proactive strategy that can reduce lead time, facilitating a more responsive system. In addition, seabasing can provide access to cargo where establishment of a base on land is not possible or very difficult due to lack of infrastructure, the presence of conflict or the lack of diplomacy within the area. Finally, a sea base can serve a role similar to an e-commerce fulfillment center by fulfilling personalized requests for cargo to be delivered to troops on the ground only when requested.

Seabasing requires conducting logistics operations from the sea. Sea-based logistics operate in a challenging environment and have unique mission characteristics. Seabasing requires increased security measures, synchronization of sea-based logistics operations with land operations, the absence of permanent infrastructure, and individual logistic transport needs. Thus, the design of sea-based logistic delivery systems is critical to ensuring rapid transfer of vital cargo stored on vessels.

The remainder of this chapter is as follows. Next, we describe the material handling environment, relating how a sea base is organized in a hierarchical structure, as well as describing the layout and material handling equipment on a specific platform, the T-AKE vessel. In Section 3, three specific seabasing scenarios including an *Iron Mountain*, *Skin-to-Skin Replenishment*, and *Tailored Resupply Packages* are described and mapped to common logistics systems and decision problems. Related literature is reviewed in Section 4. Finally, we conclude with decision-making challenges and open research questions for seabasing distribution and logistics processes.

2. Material Handling Environment

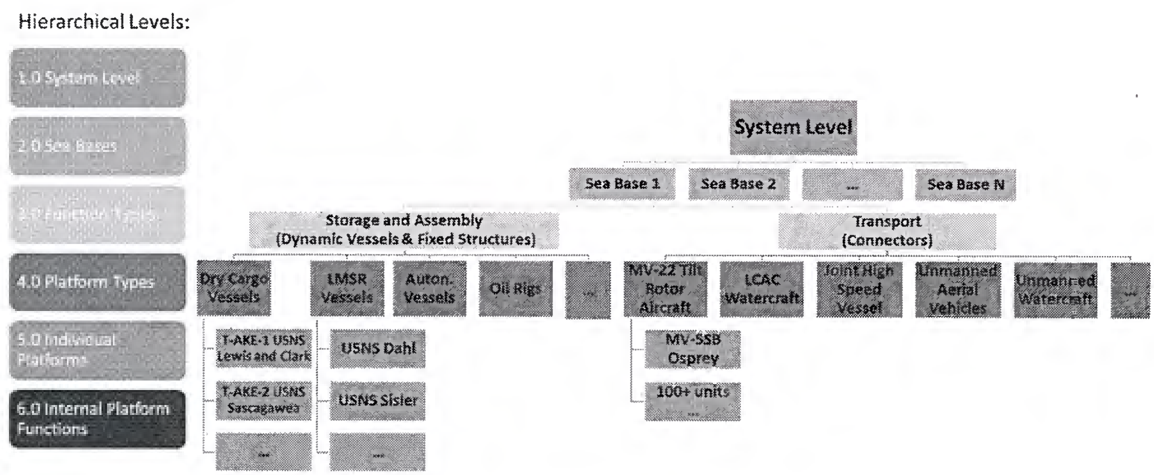


Figure 1: A sea base consists of multiple platforms organized in a hierarchical organization structure

As illustrated in Figure 1, a sea base consists of multiple platforms and hierarchical levels. Two functional types make up a sea base. The first is *Storage and Assembly*, which is achieved through dynamic vessels and static platforms; the second is *Transport*, which is achieved through connectors, which can be watercraft, aircraft, unmanned systems or other technologies.

For each of the functional types, platforms with varying capabilities and degrees of autonomy are employed. For the *Storage and Assembly* function, different vessel types are used to store and assemble resources. Examples of dynamic storage and assembly platform types include the T-AKE 1 USNS Lewis and Clarke Class Dry Cargo Vessels, the LMSR (large medium roll-on/roll-off) vessels, and autonomous vessels, among others. Static storage and assembly platform types could include abandoned oil rigs or other fixed structures. Each platform type consists of multiple individual instances of the platforms. Examples of internal platform functions include receiving, storage, retrieval, assembly, packaging, and shipping.

For the *Transport* function, a wide range of connectors with varying capabilities and degrees of automation are available. Platform types include both aerial delivery platforms (such as the MV-22 tilt rotor aircraft and unmanned aerial vehicles), as well as watercraft (like Landing Craft Air Cushion (LCAC), Joint High Speed Vessels (JHSV), and Unmanned Watercraft). For most of the connector types, there are hundreds of individual platforms in operation

2.1 T-AKE Ship Background

While many platforms can make up a sea base, T-AKE ships have been targeted for use by the US military to fulfill the *Storage and Assembly* function needs of sea bases [14]. T-AKE ships were ordered in 2001 as a way to provide quick replenishments for the United States Navy at sea without the need of having to go back on land [22, 23]. The construction of these ships was contracted to General Dynamics; the first ship is called the T-AKE 1 USNS Lewis and Clark and was finished in 2005 [7]. Since then, 13 more T-AKE ships were contracted to General Dynamics, with the last ship, T-AKE 14 USNS Cesar Chavez delivered in October 2012 [7].

While some aspects of the ships have changed throughout the 14 iterations, many of the key features have stayed the same. The T-AKEs are built to specification of 689 feet length, 106 ft beam, 29.9 ft draft, dry cargo capacities of 6,675 MT, and a design speed of 20 knots [7, 22, 23]. The ships can hold 23,450 bbl of fuel and 52,800 gal of cargo potable water [7]. A side view of a T-AKE is given in Figure 2.

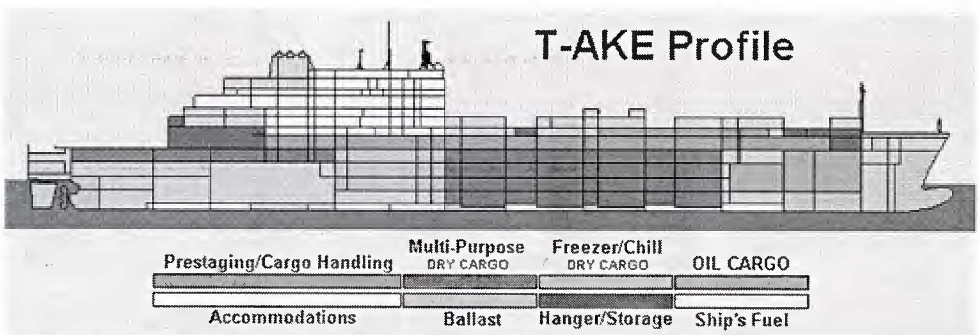


Figure 2: A side-view of the Lewis and Clarke (T-AKE 1) Dry Cargo Ship available from http://navy.memorieshop.com/T-AKE/

As described in the *T-AKE 1 Class Ship Cargo Ship and Equipment Operation and Maintenance Familiarization Crew Study Guide* [13], T-AKE ships have 5 levels, with level one being the top level above water and decreasing from level one to five as one descends lower into the ship. The levels are further divided vertically into holds. On top of the ship is a landing pad for helicopters and small airplanes. Connecting the levels are 8 elevators. On the ship are storerooms divided into categories such as parts, metal, lumber, miscellaneous, freezer, pipe, extra-large, and ones for specific chemicals that tend to be smaller in size. There are multi-purpose cargo holds on each vessel that can be reconfigurable and used to store a variety of cargo, from 8-by-20-by-8.5 feet cargo containers, quadcons, and pallets. Both multi-purpose dry stores and freeze stores are available. To increase storage density in these multi-purpose holds, cargo are typically stacked on top of each other and packed tightly with very little room for movement.

Each T-AKE ship has several types of material handling equipment to move cargo internally within the ship and within the holds [13]. There are four cargo cranes that can lift cargo of approximately 11 tons and are used for loading and unloading of cargo. For internal transport, there are 8 diesel fork lifts, as well as 14 electric fork. In addition, electric side loaders and ordnance trailers are available. Thus, each T-AKE ship contains over 30 material handling equipment with varying capabilities to be used for internal cargo transport and storage operations.

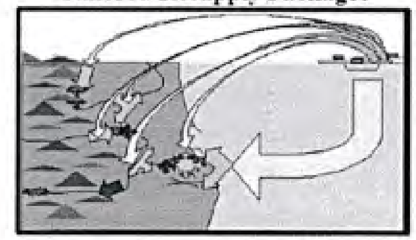
3. Sea Based Distribution Network Scenarios

In this section we discuss three different seabasing distribution network scenarios, which exhibit varying levels of complexity [14]. In Table 1, the three scenarios -- *Iron Mountain*, *Skin-to-Skin Replenishment*, and *Tailored Resupply Packages* -- are described and mapped to common logistics system characteristics and decision problems.

3.1 Iron Mountain Scenario

The first and simplest scenario is an *Iron Mountain* scenario. In this scenario, everything loaded on the vessel is offloaded, and the vessel is solely used for transportation, moving cargo loaded at a given origin to its destination, where all cargo is then offloaded. The key performance indicators are to maximize storage density and to maximize item-

Table 1: The characteristics of three different seabasing distribution network scenarios.

	Iron Mountain 	Skin-to-Skin Replenishment 	Tailored Resupply Packages 
Demand-Level	Mission-Level	Vessel-Level	Individual-Level
Demand Characteristics	Pushed	Plannable	Emergent
Types of Requests	Everything Loaded at its Origin is Offloaded at the Destination.	Bulk Requests for Standard Replenishment Items.	On-Demand, Personalized Requests.
Handling Units	Container, Vehicle, Pallet	Pallet, Case	Case, Piece
Key Performance Indicators	Maximize Storage Density and Item-Type Assortment	Maximize Storage Density and Minimize Transfer Time of Cargo Between Vessels.	Maximize Responsiveness, Item-Type Assortment, and Storage Density
Functional Requirements	Dense Storage	Dense Storage and Item-Type Segmentation	Dense Storage and Selective Offloading
Analogous System	Transportation System	Unit-Load Storage System	Order-Fulfillment System
Decision Problem	Knapsack Problems	Less-than-Truckload Loading and Routing Problems	E-Commerce Order Fulfillment Problems

type assortment. An *Iron Mountain* scenario is the simplest scenario as internal vessel logistics do not play a large role and the handling unit is at the unit load level (e.g., a container, pallet, or even a vehicle). Given all cargo are loaded and unloaded at the same time, the decision problem associated with this scenario is a knapsack problem, which identifies what type of cargo and its quantity to load, given anticipated utility of each cargo type and a limited capacity.

From a holistic supply chain perspective, an *Iron Mountain* scenario is disadvantageous as it is rigid, and does not allow for much flexibility. The items delivered in an *Iron Mountain* scenario are pushed and not pulled. Forecasting demand is required to be done when the sea base ship is loaded. From forecasting theory, as the length of time into the future that demand is being forecasted increases, the accuracy of the forecast decreases. Additionally, mission and thus demand requests can change while the ship is en route, thus making an *Iron Mountain* scenario rigid and not responsive. Given everything loaded is offloaded, this scenario does not allow for much flexibility. A requirement in this scenario is the presence and coordination of a land force that can accept and transport cargo once it is offloaded. Depending on the area where the port is located, there may be need for armed force security to protect the cargo during the offloading and transport process. An *Iron Mountain* scenario, transfers responsibility of storage, transport, and order-fulfillment of cargo to the troops on the ground, which is often in a hostile environment and has limited resources.

3.2 Skin-to-Skin Replenishment Scenario

A more complex scenario is a *Skin-to-Skin Replenishment* scenario, where sea base vessels will transfer cargo to replenish requests by other vessels in open water. This can occur during benign operation, as well as in combat missions. Replenishment using a sea base eliminates the need for all vessels requiring cargo to dock on land to restock. It is especially useful near hostile areas where replenishment is not available. Instead of having to sail back to the nearest friendly port, a vessel can be replenished with cargo for multiple vessels and the sea base can make replenishment deliveries to the other vessels. Therefore, the decision problems in a *Skin-to-Skin Replenishment* scenario are similar to those for a less-than-truckload company. Specifically, decisions about which vessels to replenish and in what order (i.e., assignment and routing decisions) are required. Such decisions are similar to a traveling salesman problem, however, an added challenge is that the delivery locations are other vessels, which have the ability to move.

The act of transferring cargo from one vessel to another vessel in open water is called skin-to-skin replenishment. There are two different ways to carry out the process. For both processes, the sea base and the vessel requesting cargo must be positioned parallel and traveling at nearly identical velocities. Typically both vessels travel at low speed in order to make coordination easier. Once this formation is achieved, one way to transport cargo is known as connected replenishment (conrep). This requires that a bridge or pulley system is used to connect the two vessels. When a bridge is used, the cargo is manually carried from the sea base to the vessel being replenished. When a pulley system is used, the cargo is shuttled via the pulley between the two vessels. Another way to achieve skin-to-skin replenishment is to use an aircraft to lift cargo from the sea base and drop it onto the flight deck of the vessel requesting cargo. This is known as vertical replenishment (vertrep). While vertrep requires less precise coordination of ship velocity than conrep, it limits the type and amount of cargo that can be transported. Also, aerial delivery vehicles, which are used for a wide range of missions are a limited resources, typically provides slower transport as opposed to a physical bridge connector. In addition, each load needs to be wrapped in special slings or cargo nets, which require additional labor and materials to prepare.

The speed in which the cargo can be transferred is an important criteria in a *Skin-to-Skin Replenishment* scenario because the two vessels are vulnerable to attack when skin-to-skin replenishment is being conducted. Given the loading of cargo impacts the speed at which cargo can be unloaded, considerations of where and how cargo are stored become important. Selectively offloading only the items needed for each vessel may be a challenge in dense storage environments, and the layout and location of cargo stored in the holds needs to be considered. Specifically, cargo needs to be organized in a way to allow for quick access to specific unit-loads.

The demand for a *Skin-to-Skin Replenishment* scenario typically occurs at the vessel-level, and thus demand requests tend to be for full pallet quantities of ordnance, dry stores, and food. Thus, a *Skin-to-Skin Replenishment* scenario requires internal flow operations similar to a unit-load warehouse. Decisions of importance are item-location allocation problems, storage layouts, and storage and retrieval operational design. Given the sea base gets advanced notice of the quantity and type of cargo requested, a *Skin-to-Skin Replenishment* scenario occurs in an environment where demand requests can be planned. Ideally, such information is provided with sufficient time to conduct strike-up processes prior to arrival of the vessel needing the replenishments. If not, questions regarding how much and what types of items should be pre-staged in anticipation of an arriving request to ensure responsiveness are of interest in this scenario.

3.3 Tailored Resupply Packages Scenario

Finally, the most complex scenario is to employ *Tailored Resupply Packages* scenario. Operationally, this requires the ability to deliver emergent requests for tailored resupply packages by selectively offloading cargo stored in the cargo holds of ships [16]. The cargo requested will be for specific items and will be requested in low units of measure (e.g., at the unit or case level). The benefit is that inventory is stored on the sea base until needed, enabling that cargo be located where and when they are most needed [12]. Being at sea also acts as a natural barrier against potential unfriendly disruption, resulting in potential security advantages. The need to support emergent requests requires forward deployed naval forces to pivot and respond in real time because needs, requirements and resource availability will change over time [15]. Therefore, response time is critical.

The internal cargo flow processes on vessels required to handle emergent requests for a *Tailored Resupply Packages* scenario can be broken down into the five functions:

(1) the transfer of cargo between vessels,
(2) the strike-down process:
(3) storage,
(4) strike-up phase, and

(5) delivery.

Similar to the *Skin-to-Skin Replenishment* scenario, the transfer of cargo between vessels occurs using either conrep or vertrep processes. The strike-down process is the transfer of cargo from the ship's flight deck to stowage spaces. Storage is done to maximize the inventory stored; consequently, pallets and containers are deeply stored, frequently stacked and often without aisles. The strike-up process is the transfer of cargo from the stowage space to the offload point of the vessel for delivery. The strike-up process is analogous to the concept of order fulfillment in the warehousing literature [18]. The strike-up process includes finding and retrieving requested items from cargo holds that are densely packed. Given this may require moving items out of the way to gain access, the process has been found to exhibit high variability [1]. After the items are retrieved and assembled, they are transported to the flight deck, where the personalized package is loaded onto a transport vehicle (e.g., high-speed vessel or aerial vehicle) for delivery to their objective location.

Given the internal cargo processes to fulfill emergent and personalized requests for cargo stored on the T-AKE 1 USNS Lewis and Clark is currently a human-intensive process, emergent and personalized requests are currently a material handling and logistics challenge. Specifically, holds that store dry cargo manually store, retrieve, and relocate pallets and containers using forklifts or pallet jacks. This is a challenging operational environment because of the dense storage environment and the need to conduct selective offloading, which requires the retrieval of specific units, perhaps even ones located in the most inconveniently placed location. This shifting can result in the growth of unit location uncertainty as the system operates. Despite an initial load-plan and initial certainty in unit locations, knowledge of unit location in dense storage environments has been observed to be lost and has resulted in time spent searching for the requested unit. For example, item location uncertainty and searching for requested crates were observed in an exercise conducted in 2012, which had a goal of observing the physical capability of ships to handle emergent requests for tailored resupply packages.

A *tailored resupply packages* scenario is the most complex scenario because emergent and personalized requests exhibit the following characteristics:

1. Requests occur as a random event in time. This results in requests being highly stochastic and time-varying.
2. Requests are highly personalized; thus, there is variability in what is requested, how it will be requested, and in its delivery location.
3. Requests are expected to be fulfilled quickly. This results in short lead time expectations. Lead time is the time from when a request is made to when the request is fulfilled.
4. Combining 1, 2, and 3 means that a wide variety of requests are made with little warning and are expected to be fulfilled quickly. This results in an environment where operational decision making is crucial.

Emergent and personalized requests result in high levels of variability in when the requests will be made, in what will be requested, in how it will be requested, and where it is needed. Variability degrades performance [5]; systems with more variability also require higher levels of inventory, capacity, and/or lead times to achieve a given service level. Also, systems with high variability exhibit high errors in forecasts, given past requests are not good predictors of future requests. The challenges associated with a *Tailored Resupply Packages* scenario are similar to the challenges faced by an e-commerce order-fulfillment system: the handling units are at the piece or case level, the system needs to be responsive, resulting in the need for quick order-fulfillment times, and last-mile delivery of personalized packages do not have the luxury of economies of scale.

4. Related Literature

In this section we review and map the existing peer-reviewed sea-based logistics research to the previously defined scenarios and logistics functions from Section 3. Focusing on the transfer of cargo from a ship to land, Kang and Gue [11] describe the concept of sea based logistics and develop a simulation model of the offloading process in an in-stream environment. Focusing on macro-level supply chain issues, Gue [8] developed an optimization model to determine the supply chain network design for distributing cargo to dispersed supply units when sea-based distribution is incorporated with land-based distribution. They find that operational levels of the location of the sea base, the amount of inventory held by combat units, the availability of transportation assets, and the timing of troop movements all play a critical role in deciding whether all deliveries should be made by air, by ground, or by air and ground distribution. Considering the use of seabasing for replenishment operations, Brown and Carlyle [3] build an integer linear program that is used to understand capacity planning needs and the impact of different operating policies. The model uses an objective to minimize shortages and maximize utilization of transport platforms and total volume delivered by a specified combat logistics force. Salmerón, Kline, and Densham [20] produced a global fleet station mission planner tool that enables the user to explore the feasibility of different missions and can be used to understand, for example, how different ship types can be used to accomplish similar missions.

The above mentioned research all takes a macro-view, and assumes cargo to be on the flight deck and does not consider internal cargo flow operations. More recently, both descriptive and prescriptive models have been developed that considers the internal cargo flow processes associated with sea basing.

Motivated by sea basing, Gue [9] studied the layout of storage systems characterized as very high density storage systems. He defined very high density storage systems as ones where not all items are immediately accessible. In such systems, shifting of other stored items to gain access to the desired item may be required. In a k-deep very high density storage system, up to $k - 1$ pallets may have to be moved 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 to retrieve the one behind it. Contrastingly, a single-deep storage system is not considered a very high density system, as all pallets are accessible directly from aisles. The highest storage density for a fixed value of the accessibility constant k is achieved in layouts that resemble the inverted T configuration [6].

Reilly, Pazour, and Schneider [19] develop descriptive models that mathematically describe the dense storage environment used in sea-based logistics system, its performance, and the relationships between factors responsible for the performance. Specifically, a very high density, class-based storage system is modeled as a discrete-time Markov chain in which the state space is the storage unit locations. The units transition from one location (state) to another based upon retrieval requests, and the movement matrix determines the likelihood of each possible location (state) change in the system. An input to this work is a retrieval algorithm, for example, the puzzle-based algorithm developed by Gue and Kim [10]. An analysis is conducted to determine the steady state probability distribution of unit locations. As the first to consider unit location uncertainty in a warehousing environment, they develop metrics that quantify and characterize unit location uncertainty. The developed model and metrics are capable of characterizing how cargo holds evolve from a highly organized state to the observed disorganized state that exhibits imperfect unit location visibility and are used to identify the impact of policies and layouts on uncertainty propagation.

Awwad and Pazour [1] study the problem of single searcher looking for a single item in a very high density storage systems with uncertainty of item locations. An inverted T k-deep storage system is considered and this work incorporates decisions about how to conduct repositioning of items that need to be moved to gain access to other more deeply stored items and their associated put-back operations, in addition to traveling, within a search procedure. An optimization model and heuristic solution approach that minimizes the expected search time for two repositioning policies are studied. 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. The search process in a very high density storage systems is shown to exhibit high variability and the full distribution of search times is recommended for downstream planning.

Scala and Pazour [21] use a value focused thinking approach with subject matter experts to evaluate how to reduce item location uncertainty in a seabasing environment. Functionally, internal cargo flow for sea-based logistics can be supported through identification technology devices, such as Radio Frequency Identification (RFID), barcoding, internal positioning systems, and camera-aided technology. These asset tracking devices are considered as alternatives to a multi-objective decision model with the goal of selecting the preferred device for seabasing logistics support. Criteria for this model include registration of inventory in the system, stowage factor enablement, storage location precision, retrieval identification accuracy, system compatibility, and security. Given the requirements of selective offloading in dense storage environments, internal positioning systems are the preferred asset tracking technology.

5. Decision-Making Challenges and Open Research Questions for Seabasing Distribution and Logistics Processes

Seabasing presents specific decision-making challenges for distribution and logistics processes. Resources to be requested from the sea base vary, and include different classes of cargo, military tactical vehicles, and personnel. Challenges from the supply-side include having limited resources, limited storage and operational space capacity. For *Skin-to-Skin Replenishment* and *Tailored Resupply Packages* scenarios, the ability to selectively offload cargo in a dense storage environment is required. Challenges on the demand-side focus on the need to fulfill emergent and personalized deliveries, which requires coordination and synchronization of sea-based logistics operations with land operations. Emergent requests can result in having imperfect visibility about when requests will be made. This results in requests being received without warning for a wide variety of resources. Thus, research is needed that (1) develops models to quantify and evaluates strategic resource allocations in a multi-level, dynamic environment, (2) analyzes and designs technologies to aid in material handling in a sea based environment, and (3) designs and evaluates operational strategies to balance the conflicting objectives of responsiveness, storage density, and asset visibility.

One of the major purposes of having a sea base is to allow for quicker response for missions; therefore, one of key characteristics of a sea base is its ability to conduct order-fulfillment processes in a prompt manner. Another is effective space utilization and the need to maximize the quantity and type of cargo on the sea base. One way to increase the quantity stored is to increase storage density. However, it has been well-established that response time and storage density are inversely related [2]. Increasing storage density can increase the amount of cargo stored. However, increasing storage density limits the amount of maneuvering space and also decreases the likelihood that all items will be directly accessible. Given all items are not directly accessible in dense storage systems, this results in increased time to response to requests, which reduces responsiveness. Given the trade-off between responsiveness and storage density, an open research question is to determine the target storage density for different environments and operational scenarios. Also, for a given storage density, optimization methods to balance the goal of minimizing lead time requirements and maximizing cargo are needed to identify promising layouts in non-depleting systems. To be responsive to changing requirements, operational policies like cycle counting and reshuffling policies [17] should be designed for very high density environments.

Automated storage and retrieval systems are capable of achieving high storage densities and accessibility of cargo. For example, many of the dense automated storage and retrieval systems are well-suited for application in sea basing. Additional research is needed to ensure that automation can function on the open sea and can address changing requirements using an automated system for logistics functions.

Given the distributed and dynamic nature of a sea base, resource allocations to determine which resources are allocated to which platform are challenging. The allocation of inventory to the platforms in the sea base should be done at the systematic level and tied to mission requirements and to product design. For example, if designs use interchangeable parts for multiple purposes and different equipment, inventory pooling effects can be leveraged and the probability of a stock out decreases for a given in-stock quantity. Additional open research questions include considering what cargo should be allocated to which platform, given that a sea base consists of a set of platforms that can be reconfigured for different missions.

Acknowledgements

This work was supported through the Office of Naval Research via a *Young Investigator Program: The Design of Responsive Sea-Based Logistic Delivery Systems, Award Number N00014-13-1-0594 and N00014-16-1-2870.* Like sea based logistics, this research would not have been possible without the contributions of a number of talented individuals, including graduate students Mohamed Awwad, Kaveh Azadeh, Shahab Mofidi, Faraz Ramtin, and Patrick Reilly; undergraduate students Kristin Elias, Catherine Ninah, Auree Postell, Corinne Skala, and Ian Shin, and faculty Debjit Roy, Natalie Scala, and Kellie Schneider.

References

[1] Awwad, M., & Pazour, J. (2015). *Search plan for a single item in an inverted T k-deep storage system*. Unpublished manuscript.

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

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

[4] Chick, S. 2012. An Introduction to Joint Operations on and from the Sea. Combined Joint Operations from the Seas Centre of Excellence, accessed August 31, 2015 at: http://www.cjoscoe.org/images/12_022-02_Encl1_CJOS_Handbook_Joint_Ops_from_the_Sea.pdf

[5] Chopra, S., and Meindl, P. (2015). Supply Chain Management: Strategy, Planning, and Operation, Prentice Hall; 6 edition.

[6] Futcher, F. W. (2003). *Selective Offload Capability Simulation (SOCS): An Analysis of High Density Storage Configurations*. Masters Thesis from the Naval Postgraduate School. Retrived from http://goo.gl/vMZtVQ

[7] General Dynamics (2016). *Lewis and Clark (T-AKE 1) Class Dry Cargo/Ammunition Ship Fact Sheet*, available at http://www.nassco.com/pdfs/T-AKE-Fact-Sheet.pdf

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

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

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

[11] Kang, K., & Gue, K. R. (1997). Sea based logistics: Distribution problems for future global contingencies. *Proceedings of the 29th Conference on Winter Simulation*, 911-916.

[12] Mabus, R. 2015. A Cooperative Strategy for 21st Century Seapower. Washington, D.C. U.S. Department of the Navy.

[13] *NASSCO, N.A., NASSCO T-AKE 1 Class Ship Cargo Ship and Equipment Operation and Maintenance Familiarization Crew Study Guide* STTAR Corp, Technical Report, Date Unknown.

[14] National Research Council (2005), Sea Basing: Ensuring Joint Force Access from the Sea, Committee on Sea Basing, DOI: 10.17226/11370, available at: http://www.nap.edu/catalog/11370/sea-basing-ensuring-joint-force-access-from-the-sea

[15] O'Rourke, R. (2015). China Naval Modernization: Implications for U.S. Navy Capabilities. Background and Issues for Congress. Washington, D.C.: Congressional Research Service. CRS Report 7-5700, RL33153, 28 July. http://news.usni.org/wp-content/uploads/2015/08/RL33153_4.pdf, retrieved 21 February 2016.

[16] Parsons, D. (2013). Marine Corps Struggles with Sea-Based Supply Lines. *National Defense Magazine*, accessed April 21, 2016 at: http://www.nationaldefensemagazine.org/archive/2013/march/pages/marinecorpsstrugglesswithsea-basedsupplylines.aspx

[17] Pazour, J. A., and Carlo, H. J., (2015) "Warehouse Reshuffling: Insights and Optimization," *Transportation Research Part E: Logistics and Transportation Review*, 73, 207-226.

[18] Pazour, J. A. and Meller, R. D., (2011). "An Analytical Model for A-Frame System Design," *IIE Transactions* 43, 739—752.

[19] Reilly, P., Pazour, J., & Schneider, K. (2015). *Propagation of unit location uncertainty in dense storage seabasing environments.* Unpublished manuscript.

[20] 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.

[21] Scala, N., and Pazour, J. A., (2016) "A Value Model for Asset Tracking Technology to Support Naval Sea-based Resupply," *Engineering Management Journal* (Special Issue on Engineering Management in the Military).

[22] United States Navy, (2015). *Dry Cargo/Ammunition Ships T-AKE* United States Navy Fact File US Navy. Accessed May 26, 2016, available at: http://www.navy.mil/navydata/fact_display.asp?cid=4400&tid=500&ct=4

[23] United States Navy, (2015). *US Navy Program Guide 2015, accessed May 26, 2016 available at http://www.navy.mil/strategic/top-npg15.pdf*

Chapter 2: Propagation of Unit Location Uncertainty in Dense Storage Seabasing Environments

This chapter has been published in the following peer-reviewed journal:

Reilly, Patrick, Pazour, Jennifer A., and Schneider, Kellie, 2017, "Propagation of Unit Location Uncertainty in Dense Storage Seabasing Environments," *International Journal of Production Research*, 55(18), 5435-5449.

Propagation of Unit Location Uncertainty in Dense Storage Environments

Dense storage systems provide high-space utilization; however, because not all units are immediately accessible, selectively offloading units can require shifting of other stored units in order to access the requested unit. Given an initial certainty in unit location, a discrete time Markov Chain is developed to quantify the growth of unit location uncertainty as a function of retrieval requests. As the first to mathematically model uncertainty propagation in dense storage operations, metrics are developed to analyze the model. A theoretical understanding of the relationship among storage density, retrieval times, and unit location uncertainty is provided. Finally, a case study using inventory and load plan data from a military application illustrate how the developed models can be used by managers to evaluate selective offloading policies and layouts.

Keywords: logistics; warehousing systems; storage systems; naval logistics; item location uncertainty;

1. INTRODUCTION

Seabasing is a maritime operation in which cargo stored on ships is delivered ashore – effectively conducting logistics operations from the sea. Strategically, seabasing provides flexibility to conduct a wide range of missions, including humanitarian aid distribution, crisis prevention, combat operations, and sustainment of military forces on the ground (Chick, 2012). Currently, seabasing is used for *Skin-to-Skin Replenishment* scenarios in which ships transfer cargo to replenish requests by other ships in open water (Pazour and Shin, 2016). Of interest, is to employ a *Tailored Resupply Packages* scenario in which emergent and personalized requests for resupply packages are delivered directly to forward deployed forces (Parsons, 2013).

A *Tailored Resupply Packages* scenario requires different operational requirements (Allen, 2006; Kemp, 2008). The ships will receive orders requesting specific units and quantities to be retrieved from the dry cargo holds. These requests will occur for a wide range of products, often with little warning. The storage systems in the dry cargo holds of the ships are designed to maximize the inventory stored; pallets and containers are packed in tightly, frequently stacked and often without aisles. The cargo holds can be characterized as very high density storage systems, in which all units are not immediately accessible (Gue, 2006).

Executing a *Tailored Resupply Packages* scenario is challenging because it requires selectively offloading cargo from very high density storage systems. The internal cargo processes currently use a human-intensive process to manually store, retrieve, and relocate pallets and containers using forklifts or pallet jacks. Retrieval operations often require relocation movements of units not requested in order to access the desired unit. Each ship is provided with a load plan when followed results in initial certainty in unit locations. Logistics systems have limited visibility aboard ships (Kemp, 2008). The ability to locate specific units within the system can be lost over time. During a recent military exercise to observe the capability of ships to handle emergent requests for *Tailored Resupply Packages*, (Sullivan, 2012), unit location uncertainty and searching for requested crates was observed. Therefore, we explore the propagation of unit location uncertainty as retrievals are made in dense storage environments, seeking answers to questions about why it occurs, how it behaves and how, if possible, it can be managed through different policies.

Asset tracking is challenging because a large number of units are stored in a small area and no fixed locations (like those used in aisle-based storage systems) for license plates exist. Radio Frequency Identification (RFID) tagged units do not identify cargo locations to the granularity needed for a responsive *Tailored Resupply Packages* scenario (Scala and Pazour, 2016). RFID can only identify the hold in which the unit is located – not the coordinate location values of the cargo. Also, combining RFID with a hand-held reader with coordinate capabilities is not feasible in a dense storage environment without reducing the storage factor. Additionally, the storage environment is a metal box structure; signal and battery life can be an issue.

The goal of this work is to mathematically understand and analyze the propagation of uncertainty in very high dense storage systems. A descriptive model for the propagation of uncertainty in unit locations in dense storage environments requiring selective offloading capabilities is developed. Given an initial certainty in unit location, the growth of uncertainty as a function of the number of retrieval requests is quantified using a discrete-time Markov Chain. This model provides a mathematical foundation to describe the observed behavior in dense storage operations and allows us to understand why location uncertainty occurs, and to quantify the loss of information overtime with different operational policies. As the first to model and analyze unit location uncertainty propagation in very dense storage operational environments, we also contribute by adding a new dimension to the warehousing and distribution literature. To do so, metrics are defined to characterize uncertainty in unit location. The developed model and metrics are used to evaluate selective offloading policies and layouts and to describe the relationship between expected retrieval times and uncertainty metrics. A case study with inventory and load plan data from a naval application is presented.

The descriptive model developed in this research provides an analytical baseline to measure and evaluate trade-offs in return on investment studies for more sophisticated material handling and asset tracking systems. Inventory inaccuracy research has shown prescriptive models accounting for uncertainty in decision making are able to achieve similar benefits as investments in tracking technology (Chen and Mersereau, 2015; DeHoratius, et al., 2008). Quantifying uncertainty in item location when selectively offloading is required in dense storage environments results in a better understanding of this uncertainty which can be used to quantify decision trade-offs.

While this work is motivated by seabasing, the model and results are general and applicable to other high density storage environments with selective offloading. Applications include "last mile" distribution facilities used for quick response in highly populated neighborhoods, unit storage and retrieval used in space exploration, e.g., aboard the International

Space Station, and procurement of medical supplies by clinical staff in healthcare operations. In addition to seabasing, unit location uncertainty has been documented in other operational contexts including retail (Camdereli and Swaminathan ,2010; Chen and Mersereau, 2015; Chuang and Oliva, 2015; DeHoratius and Raman, 2008; Kang and Gershwin, 2005), construction (Ergen, Akinci et al., 2007; Song, Haas et al., 2006), supply chains (Zhang, Goh et al., 2011), and the military (National Academies of Science, 2014; Yopp, 2006).

2. REVIEW OF RELEVANT LITERATURE

A number of peer-reviewed studies have been conducted on seabasing operations. The majority focus on macro-level supply chain issues and assume cargo is on the flight deck (ignoring internal cargo movement and storage). Kang and Gue (1997) build a simulation model to evaluate the performance of in-stream offloading of containers. Gue (2003) develops an optimization model to determine the supply chain network design for distributing cargo to mobile supply units when seabasing is incorporated with land-based distribution. Brown and Carlyle (2008) develop an integer linear program to minimize shortages and maximize utilization of transports by a specified combat logistics force. Salmerón, Kline et al. (2011) create a tool to improve humanitarian missions through the optimization of routes and schedules.

While there is no shortage of research on warehouse operational problems (Gu, Goetschalckx et al., 2007; Gu, Goetschalckx et al., 2010), all of the existing warehouse decision making models assume unit locations are known with certainty. Our work compliments the existing research on dense storage systems. Futcher (2003) uses simulation to evaluate mean retrieval times in high density storage systems, with different configurations, densities, retrieval policies and demand scenarios. Gue (2006) analyzes very high density storage system layouts and determines layouts maximizing storage density for a given accessibility factor. Gue and Kim (2007) develop retrieval algorithms for minimizing the number of moves to retrieve a unit in high density, puzzle-based systems. Gue, Furmans et al. (2014) describe a puzzle-based system with decentralized control. The literature in other dense storage applications has focused in areas such as compact automated storage and retrieval systems (Vasili, Tang et al., 2012; Yu and de Koster, 2009), block stacking environments (Derhami et al., 2016; Jang et al., 2013), and container terminal storage environments (Carlo, Vis et al., 2014). All of these dense storage papers assume unit locations are known with certainty at all times.

Literature on inventory imperfect visibility includes empirical studies (DeHoratius and Raman, 2008); analytical models accounting for imperfect visibility when making inventory decisions (Chen and Mersereau, 2015; Kang and Gershwin, 2007) and when assessing actions to decrease inaccuracies (Kok and Shang, 2014). There is extensive literature on the field of search theory, where the need to search for targets when their location is not known with certainty is considered (Alpern, 2012; Chung and Burdick, 2012; Kadane, 2015; Shechter, Ghassemi et al., 2015). The search effort is evaluated in terms of expended resources, like manpower and time. Generally in search theory, an underlying probability distribution describes the likelihood of the target being found in particular places within the system. This probability distribution is typically given. We are aware of no work considering the origin and subsequent growth of uncertainty in the system.

This study uses Markov chains to model uncertainty propagation throughout a system. Forming a large body of literature, (e.g., Karlin ,2014; Puterman, 2014), Markov chains are used to quantify the level of uncertainty in a system as it transitions between states. Markov chains have wide applications, including military personnel dynamics (Zais and Zhang, 2016), disease progression (Craig and Sendi, 2002), and inventory policies (Larsen and Turkensteen, 2014). However, Markov chains have not yet been used in dense storage systems.

3. MODEL TO DESCRIBE UNIT LOCATION UNCERTAINTY

Markov chains are used to quantify the level of uncertainty over time as a system transitions between states (Puterman, 2014). In this section, the changing uncertainty in unit locations in a very high density environment is captured and analyzed through the development of a discrete-time Markov chain. The state space for this Markov chain is given by the physical storage locations, and the transition probability matrix is developed based on a given retrieval policy and the demand for requested units. The probability distribution for the initial states is based on the unit's initial locations, which are assumed known and given. The retrieval history for requested units is unknown and retrieval requests are assumed to be independent of one another. Thus, the Markov principle holds as a unit's location after the next retrieval is based only on its current location.

Consider a very high density storage system subject to selective offloading. Let L denote the state space, which is the set of storage locations such that $l = 1, 2, \dots, n$ map to the physical spaces where individual units (e.g. pallets or containers) are stored. Requests are made to retrieve specific units, and the system is observed during these discrete points in time. Let T denote the set of retrieval requests $t = 0, 1, \dots$, where t denotes the period prior to the $(t + 1)^{\text{th}}$ retrieval. A requested unit may not be immediately accessible. Therefore, retrieval operations may require shifting of other stored units to access the requested unit. This shifting can result in uncertainty in a unit's location over time. Let $\boldsymbol{W_t}$ be a discrete random vector denoting the locations of units in the set of storage locations after t retrievals. The elements of the random vector $\boldsymbol{W_t}$ are ${w_t}^l$, which are random variables denoting the location of the unit originally in location l after t retrievals. Further, let z_t be a random variable indicating the location to be accessed after t retrievals.

Suppose this very high density storage system operates under a class-based storage policy. That is, the system is partitioned into classes of locations strictly dedicated to some particular group of units. Within a given storage class, units are randomly assigned storage locations and the retrieval operations for separate classes of units are independent of one another. Class-based storage is common in traditional retail storage environments because of its ease of implication and reduction in travel times when units are partitioned by demand. The cargo hold of a naval ship may contain ammunition, medical supplies, and non-perishable food. Each of these inventory types can represent a storage class. Each storage class is allocated to a specific storage area of the cargo hold, and the specific units are found randomly within their assigned storage class.

Let J denote the set of storage classes, $j = 0, 1, \dots, |J|$, and suppose each storage class, C_j, contains a subset of the locations in L, i.e. $C_j = \{l \in L | l \text{ is of class } j\}$. Locations may only be assigned to exactly one storage class, thus $\bigcap_{j \in J} C_j = \emptyset$ and $\bigcup_{j \in J} C_j = L$. Furthermore, the units designated to a particular class are stored randomly in locations within the class. Units assigned to a particular storage class may move to any location within that class; thus, each storage class is said to be a *communicating* class (Ross, 1996).

Let Δ_j denote the probability that a retrieval request is made for a unit in storage class C_j and let x be the requested unit, where x is a random variable. We compute the probability that unit x is in storage location l by conditioning on the storage class to which unit x is assigned:

$$\Pr(x = l) = \sum_{j \in J} \Pr(x = l | x \in C_j) \Pr(x \in C_j) \tag{1}$$

Since $\bigcap_{j \in J} C_j = \emptyset$, then

$$\Pr(x = l) = \Pr(x = l | x \in C_j) \Pr(x \in C_j) \quad \forall l \in C_j \tag{2}$$

The individual units within a given class are equally likely to be requested for retrieval. Therefore,

$$\Pr(x = l) = \frac{1}{|C_j|} \Delta_j \quad \forall l \in C_j, j \in J \tag{3}$$

Thus, $\frac{\Delta_j}{|C_j|}$ is the probability of a request for the storage unit that originated in location l.

The storage system considered in this model is non-depleting, meaning the storage units never leave the system. Each storage location contains only one type of unit at any time and it is assumed that no retrieval request will demand the entire contents of the unit load. This represents, for example, case-level requests from pallets. The pallet containing the requested unit will be accessed, cases will be retrieved from the pallet, and the pallet with the remaining cases will be returned to the storage system, although possibly in a new location. Thus, the propagation of uncertainty is analyzed through a series of retrieval operations only. The storage locations remain stationary while the storage units move between different locations when retrievals occur. Due to insufficient asset tracking, the unit retrieval history is not kept and is, thus, unknown. Therefore, when a retrieval request is made, the request is made denoting the unit's initial location.

2.1 Transition Probability Matrix

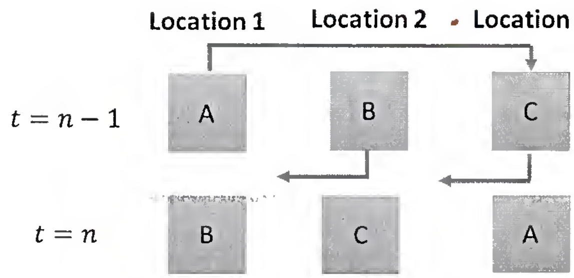
The probabilities associated with the location of each unit must be expressed and updated after each retrieval. The location probability matrix $\boldsymbol{P}^{(t)}$ is the t-step transition matrix, which provides the probability of each storage unit being in each storage location within the system after t retrievals. Therefore, an individual entry $p_{l\hat{l}}^{(t)} \in \boldsymbol{P}^{(t)}$ is the probability that the unit originally in location l is in location $\hat{l}$ after t retrievals. Initially at $t = 0$, all of the unit locations are known with certainty. Without loss of generality, we assume the first unit starts in location one, the second in location two, and so on. Thus, prior to the first retrieval the location probability matrix $\boldsymbol{P}^{(0)}$ is equivalent to the identity matrix.

The movements of storage units in response to a retrieval request are dictated by a retrieval policy. The location probability matrix $\boldsymbol{P}^{(t)}$ is updated based on the current state of the system for a given retrieval policy and demand distribution. We assume retrieval policies are unambiguous, all possible movements are clearly defined and their probabilities are known. The retrieval policies can be expressed mathematically via the retrieval matrices. For each location in the storage system, the retrieval matrix expresses the movements executed when that location is accessed. Suppose unit x is requested for retrieval, and let $\hat{l}$ denote the location in which unit x was found. Thus, retrieval matrix $\boldsymbol{R}_{\hat{l}}$ is an $|L| \times |L|$ matrix used to express the movements executed when location $\hat{l}$ is accessed for the retrieval of requested unit x. Note, each location in $\boldsymbol{L}$ is assigned a retrieval matrix $\boldsymbol{R}_{\hat{l}}$ and each entry $r_{ik}^{\hat{l}}$ in $\boldsymbol{R}_{\hat{l}}$ is the probability that the unit in location i moves to location k when location $\hat{l}$ is accessed for a retrieval. Because we have a non-depleting system, one (and exactly one unit) is required to be in each of the $l \in L$ locations before a retrieval and one (and exactly one unit) is required to be in each of the $\hat{l} \in L$ after a retrieval. Therefore, for a retrieval matrix to be valid,

$$\sum_{i=1}^{n} r_{ik}^{\hat{l}} = 1 \; \forall \, k \in L \, \forall \hat{l} \in L \tag{4}$$

$$\sum_{k=1}^{n} r_{ik}^{\hat{l}} = 1 \; \forall \, i \in L \, \forall \hat{l} \in L \tag{5}$$

Both pure and probabilistic retrieval strategies are accommodated in the model. A pure strategy is one in which each location has only one possible reconfiguration associated with it, and it is executed with certainty when that location is accessed for a retrieval. A probabilistic strategy is one in which a retrieval operation can be executed in more than one way but the probabilities of each occurring is known. Consider, for example, the pure strategy provided in Figure 1. Units A, B, and C occupy three locations. A pure policy is for any unit accessed for a retrieval to move to the end of the row (Location 3) and all units previously to the right of the retrieved unit shift left to fill the gap. Figure 1 provides the retrieval matrix given that the unit in location $\hat{l} = 1$ is selected for offloading, and a graphical representation of the associated movement of units in locations 2 and 3.



$R_{\hat{l}=1}$

From/To	1	2	3
1	0	0	1
2	1	0	0
3	C	1	0

Figure 1. Movement and Retrieval Matrix for Example Policy

A retrieval policy must be defined to describe what will happen to all the other units in the system, if a unit in location $\hat{l} \in L$ is accessed for retrieval. Thus, a retrieval policy is expressed explicitly through a set of retrieval matrices (i.e., there is a movement matrix $\boldsymbol{R}_{\hat{l}}\ \forall\ \hat{l} \in L$). However, the retrieval policy does not provide information regarding the probability that a requested unit is found in location $\hat{l}$. Therefore, let d_l denote the probability that a retrieval request is made for the unit originally stored in location l. Recall from Equation (3)

$$d_l = \frac{\Delta_j}{|C_j|}\ \forall\ l \in C_j, j \in J \tag{6}$$

Recall z_t is a random variable indicating the location to be accessed after t retrievals. Because units within a storage class have a uniform demand and the system is non-depleting, the probability of retrieving a requested unit from location $\hat{l}$ is equally likely for each unit within a storage class. Furthermore, if the system is comprised of a single storage class, the probability of retrieving the requested storage unit from location $\hat{l}$ is a constant value, i.e.

$$Pr(z_t = \hat{l}) = d_{\hat{l}} = d_l = \frac{\Delta_j}{|C_j|} = \frac{1}{n} \tag{7}$$

For systems with multiple storage classes, we consider a set of $|C_j|$ individual systems each with its own storage class. The likelihood of location $\hat{l} \in C_j$ being accessed for a retrieval also remains at a constant value:

$$Pr(z_t = \hat{l}) = \begin{cases} \frac{\Delta_j}{|C_j|} & \text{if } \hat{l} \in C_j \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

The one-step transition matrix $\boldsymbol{P}^{(1)}$ is computed based on the expected movements executed in the system. The expected movements executed can be calculated probabilistically by combining the retrieval matrixes, which provides the movements made when a location is accessed, with the probability a location is accessed.

$$\boldsymbol{P}^{(1)} = \sum_{\hat{l}=1}^{n} Pr(z_t = \hat{l}) R_{\hat{l}} \tag{9}$$

The one-step transition matrix is expanded to illustrate the stationary assumption holds.

$$P^{(1)} = \sum_{j\in J}\sum_{\hat{l}\in C_j} Pr(z_t = \hat{l})R_{\hat{l}} = \sum_{j\in J}\frac{\Delta_j}{|C_j|}\sum_{\hat{l}\in C_j} R_{\hat{l}} \quad (10)$$

The unambiguous retrieval policy, expressed per location by $R_{\hat{l}}$, remains constant, thus the one-step transition probability remains constant for all t.

2.2 Definition and Properties of the Discrete-Time Markov Chain

The very dense, class-based storage system is defined as a Discrete-Time Markov chain. Let $\{W_t, t = 0,1, \dots\}$ be the Markov Chain, where ${w_t}^l$ is an element of the random vector W_t denoting the location of the unit originally in location l after t retrievals. The state space is the storage unit locations. The probability that each unit is in any location can be quantified after any number of retrievals through the t-step probability matrix of our Markov chain, denoted as $P^{(t)}$. Note, $Pr({w_t}^l = \hat{l})$ is equivalent to an individual entry $p_{l\hat{l}}^{(t)} \in P^{(t)}$ and is the probability that the unit originally in location l is in location $\hat{l}$ after t retrievals, which can be computed recursively.

$$P^{(t)} = Pr({w_t}^l = \hat{l}) = \sum_{\hat{l}\in L}\sum_{k\in L} p_{l\hat{l}}^{(t-1)} r_{lk}^{\hat{l}} Pr(z_t = \hat{l}) \;\; \forall l \in L; \hat{l} \in L; t > 0 \quad (11)$$

The likelihood of each possible location change in the system is based upon a retrieval policy and the demand, captured in the one-step transition matrix $P^{(1)}$. The probability distribution of the next state depends only on the current state and not the sequence of events preceding it Initially, at $t = 0$, the probability distribution is $P^{(0)}$and is the identity matrix.

The Markov Chain is reducible by storage classes. All states in a given storage class are accessible to all other states in the same storage class C_j. Thus, for a given storage class, the locations $l \in C_j$ are irreducible, aperiodic (given assumption of uniform demand) and positive recurrent.

3.3 Steady-State Probability Distribution

By the Fundamental Theorem of Markov Chains, any irreducible, aperiodic, positive recurrent Markov Chain has a unique steady state distribution π (Puterman 2014). Thus, each storage class C_j in the very dense system has a unique stationary steady state distribution π_j. Let $\pi_{j\hat{l}}$ denote the steady state probability of any unit in class C_j being in location $\hat{l}$. The steady state distribution for each storage class is given by:

$$\left\{\pi_j \;\middle|\; \pi_j P^{(1)} = \pi_j \, ; \sum_{\hat{l}\in C_j}\pi_{j\hat{l}} = 1\right\} \;\forall j \in J \quad (12)$$

For any valid set of retrieval matrices, $\mathbf{P}^{(1)}$ is always doubly stochastic. The assumption of uniform demand within storage classes results in a constant, doubly stochastic one-step probability transition matrix. Therefore, the steady state probability distribution π_j will be uniform according to the number of locations in class C_j.

By the Fundamental Theorem of Markov Chains, the uniform solution must also be unique for each class. The steady state probabilities are:

$$\pi_{j\hat{l}} = \frac{1}{|C_j|} \; \forall \, \hat{l} \in C_j, \forall \, j \in J \tag{13}$$

A unit is equally likely to be anywhere in the storage class. The steady state probabilities of $\boldsymbol{\pi}_j$ for locations in C_j are equal, and zero for locations in other classes; thus, $\pi_{jl} = \pi_{j\hat{l}}$ when both l and $\hat{l}$ are in C_j. When the system is comprised of a single storage class, the steady state probability of any unit being in any location $\hat{l}$ simplifies to:

$$\pi_{\hat{l}} = \frac{1}{n} \; \forall \, \hat{l} \in L \tag{14}$$

For any set of communicating location states, knowledge dissipates with retrievals and steady state is its complete absence. This loss of storage unit location certainty has been directly observed in seabasing operations (Sullivan 2012). We now have a mathematical model capable of quantifying this phenomenon. This model, combined with metrics defined in Section 4, is then used to evaluate trade-offs associated with different operational policies in Section 5.

3. Metrics to Measure Uncertainty Behavior in Dense Storage Environment

Until now, unit location uncertainty has not been explored in existing warehouse literature; therefore, we define seven metrics to measure uncertainty behavior in dense storage environments. These metrics, in addition to the common performance metrics of expected retrieval effort and storage density, are used to provide insights into dense storage operations.

First, the steady state probability of any unit in class C_j being in location $\hat{l}$, $\pi_{j\hat{l}}$, is called the *limiting probability*. Second, the one-step transition probability that a unit in location l will not move from that location with the next retrieval, $p_{ll}^{(1)}$, is called the *stationary location likelihood*. Third, the *original location likelihood*, U_{tl_0} is defined as the probability a unit is found in location l_0 after t retrievals, where l_0 is the location containing the unit at $t = 0$.

$$U_{tl_0} = p_{l_0 l_0}^{(t)} \; \forall t \in T; \; \forall l_0 \in C_j \; \forall j \in J \tag{15}$$

Fourth, *location stability*, σ_l, evaluates the likelihood of possible movements a unit may make in a storage class from a given location $l \in C_j$.

$$\sigma_l = \sqrt{\frac{\sum_{l' \in C_j} (p_{ll'}^{(1)} - \bar{m}_l)^2}{|C_j| - 1}} \; \forall \, l \in C_j \; \forall \, j \in J \tag{16}$$

where $\bar{m}_l$ is the average of the lth row of the one-step transition matrix if $l \in C_j$ in $\boldsymbol{P}^{(1)}$. This metric will be zero if all values are identical and represents a large number of movements are possible from that location, resulting in an unstable location. The highest possible value occurs when one entry is 1 and the rest are zeros, implying a high level of stability.

The fifth metric is *retrievals to steady state*, RSS_l, which denotes the number of retrievals to the system for location $l \in C_j$ to approach steady state behavior. Approaching steady state more quickly means the level of certainty of a unit's location is lost more quickly. A location l is considered to have approached steady state if the probabilities of its original contents being in each location in class C_j all fall within a given value ε: $\frac{1}{|C_j|} \pm \varepsilon$. The value of ε would vary based on system size and desired accuracy. In the experiments that follow $\varepsilon = 0.05$ was used.

Our sixth metric, *mean retrievals to steady state*, μ_{RSS} is a measure of the overall system performance. For any dense storage system, it is the mean of all RSS_l for every location.

$$\mu_{RSS} = \frac{1}{|L|} \sum_{l=1}^{|L|} RSS_l \tag{17}$$

To locate the unit originally in location $l \in C_j$, $E(SL_l)$ denotes the *expected locations to search* in steady state. To see how multiple locations containing the same contents can affect the expected number of locations to search, let the number of copies of the unit that originated in location l be designated as k_l. Because the system has reached steady state, the k_l copies of the unit that originated in location $l \in C_j$ are equally likely to be found anywhere in the $|C_j|$ locations. The probability of each location $l \in C_j$ containing one of the k_l copies is $\frac{k_l}{|C_j|}$.

$$E(SL_l) = \sum_{l'=1}^{|C_j|-k_l+1} l' \frac{\binom{|C_j|-l'}{k_l-1}}{\binom{|C_j|}{k_l}} = \frac{|C_j|+1}{k_l+1} \; \forall l \in C_j \; \forall j \in J \tag{18}$$

The full proof is in Appendix A. When $k_l = 1$, the expression simplifies to $\frac{|C_j|+1}{2}$, which is the expected value of the uniform distribution with a maximum of $|C_j|$ and minimum of one.

4. Numerical Examples and Insights

We provide numerical examples using our developed model to evaluate the impact different selective offloading, training, and layouts policies have on location uncertainty metrics.

Three selective offloading policies using a puzzle-base system are evaluated. A puzzle-based system consists of equally-sized locations that can each move independently in any direction. The first policy is a pure strategy (denoted as the *puzzle policy*) outlined by *Algorithm 1* in Gue and Kim (2007). The *puzzle policy* determines how to move each unit to retrieve a selected unit with known location from a high-density, puzzle-based storage system with a single open location and a single input/output (I/O) point. The algorithm is based on using the open location as an escort, which can create an open path from the requested item to the I/O point. Paths to the I/O can be accomplished with two types of escort moves, one that moves the requested item diagonal of the escort location and takes 3 moves; another one that moves the requested item to a location opposite the escort location and takes 5 moves. The algorithm begins creating the path in a direction that maximizes the number of 3-moves. The *puzzle policy* is an optimal retrieval policy when the objective is to minimize the average number of unit movements required.

The second policy is a pure strategy in which the initial movements of each unit are in the direction opposite of the puzzle policy (i.e., switch lines 2 and 6 of *Algorithm 1* in Gue and Kim (2007)). This policy, denoted as the *alternative policy*, is suboptimal in terms of minimizing the number of unit movements required to retrieve a unit because the escort square starts creating the open path in the direction that maximizes 5-moves rather than 3-moves.. The third policy (denoted as the *mixed policy*) is a probabilistic strategy intended to emulate training error. This policy is a composite of the two pure strategies in which the *puzzle policy* is executed 80 percent of the time and the *alternative policy* 20 percent of the time.

Seven system configurations, each with a different number of storage locations and ratio of length to width, are tested. Each rectangular layout has the I/O point in the lower left hand corner and has five columns of storage locations. The configurations vary the number of rows from 2 to 8. The locations l are numbered in ascending order beginning with the location to the right of the I/O point, as shown for class A in Figure 3. Initially each test configuration consists of only one communication class. In Section 5.3–5.5, multiple communication classes are considered.

The experiments were conducted using Microsoft Excel. Specifically, the one-step transition matrix $\boldsymbol{P}^{(1)}$ for each of the three selective offloading policies applied to each of the seven system configurations were calculated and can be found in Reily (2015). The numerical results either (1) apply the definition of the metrics directly from the one-step transition matrix (i.e., the *stationary location likelihood,* $p_{ll}^{(1)}$, *and* the *location stability*, σ_l), (2) are calculated using derived results (i.e., *limiting probability* is calculated using Equation (13) and the *mean retrievals to steady state*, μ_{RSS} is calculated using equation (17)), or (3) numerically determined by using Excel matrix operations of the one-step transition matrix (i.e., *original location likelihood*, U_{tl_0} is calculated based on the t step transition matrix using Equation (15); while the *retrievals to steady state*, RSS_l, are determined by calculating the metric for a single retrieval request, then two requests, three, and so on until the location l metric is within ε of steady-state performance). Detailed experimental results are also provided in Reily (2015).

5.1 Evaluation of Selective Offloading Policies

First, the impact of different selective offloading policies on the system-wide uncertainty measure, μ_{RSS}, is evaluated. The mean retrievals to steady state is calculated as the average across all locations for a given policy in a given system. Large values of μ_{RSS} represent systems where unit location knowledge is maintained longer. For the seven layouts and the three selective offloading policies, Table 1 displays the average movement effort and μ_{RSS} with $\varepsilon = 0.05$. The average movement effort (AME) is the mean number of unit moves required to retrieve a unit with a known location, and recall n indicates the number of storage locations in the system.

Table 1: Average movement effort (unit moves) and μ_{RSS} by selective offloading policy

Configurations	n	Puzzle Policy		Alternative Policy		80/20 Mixed Policy	
		AME	μ_{RSS}	AME	μ_{RSS}	AME	μ_{RSS}
2x5	9	10.1	8.7	10.8	9.0	10.2	8.7
3x5	14	11.4	19.5	12.3	20.2	11.6	18.7
4x5	19	13.2	26.7	14.2	29.4	13.4	23.8
5x5	24	15.3	31.2	16.3	32.4	15.5	27.8
6x5	29	17.7	32.3	18.8	37.5	17.9	30.7
7x5	34	20.2	34.5	21.4	42.3	20.4	33.4
8x5	39	22.9	36.8	24.1	47.4	23.1	35.8

Different selective offloading policies affect how quickly knowledge of unit location is lost. The *alternative policy* consistently yields greater mean retrievals to steady state than the *puzzle policy*, a positive result, yet also consistently has a greater expected retrieval effort, a negative result. The *puzzle policy* is optimal in terms of average movement effort when unit location certainty is known (Gue and Kim, 2007); thus, a performance tradeoff exist:

> *Insight 1: Optimality in terms of minimizing expected movement effort assuming known unit location information does not imply optimality in terms of uncertainty management.*

The mixed policy is an inefficient solution in terms of the two objectives. The average movement effort of the mixed policy is a weighted sum of the puzzle and alternative policies, and the average movement effort of the mixed policy falls between that of the puzzle and alternative policies for all seven configurations. The mixed policy performs the worst in terms of uncertainty management for all seven configurations as measured by μ_{RSS}. The convolution of pure strategies (which can occur due to lack of training and/or standardized policies) adds additional uncertainty to the system, reducing the time to approach steady state.

> *Insight 2: For a given storage configuration, selective offloading policies and the level of standardized procedures impact uncertainty measures. Operational policies impact the unit location uncertainty.*

5.2 Evaluation of Location Performance

The metrics σ_l, RSS_l, U_{tl_0} and $p_{ll}^{(1)}$ are used to describe the behavior of individual locations in a dense storage operation. These uncertainty metrics are displayed along with two common metrics used to evaluate policies in operations with known unit location in Table 2 for the 3-by-5 system. The original location likelihood after $t = 10$ retrievals, U_{10l_0}, is reported. The movement effort to retrieve a unit known to be in location l to the I/O point is denoted as E_l. Each location's distance to I/O (I/O_d) is the number of spaces from each of the locations to the I/O point measured using a rectilinear distance metric.

Table 2: Location measures of a 3x5 configuration for the selective offloading policies.

		Puzzle Policy					Alternative Policy					Mixed (80/20) Policy				
l	I/O_d	E_l	σ_l	RSS_l	U_{10l_0}	$p_{ll}^{(1)}$	E_l	σ_l	RSS_l	U_{10l_0}	$p_{ll}^{(1)}$	E_l	σ_l	RSS_l	U_{10l_0}	$p_{ll}^{(1)}$
1	1	1	0.111	12	0.070	0.214	1	0.160	15	0.069	0.143	1	0.115	14	0.070	0.200
5	1	1	0.150	16	0.080	0.500	1	0.160	14	0.073	0.143	1	0.144	16	0.077	0.429
2	2	7	0.127	10	0.071	0.429	7	0.162	11	0.072	0.429	7	0.131	10	0.070	0.429
6	2	5	0.124	13	0.083	0.429	5	0.129	12	0.071	0.143	5	0.115	14	0.078	0.371
10	2	7	0.183	19	0.168	0.714	7	0.219	29	0.312	0.857	7	0.19	20	0.180	0.743
3	3	13	0.169	15	0.095	0.643	13	0.175	10	0.088	0.643	13	0.17	14	0.095	0.643
7	3	9	0.118	9	0.074	0.429	11	0.111	9	0.069	0.357	9.4	0.112	9	0.071	0.414
11	3	9	0.169	21	0.154	0.643	11	0.219	29	0.312	0.857	9.4	0.178	22	0.158	0.686
4	4	19	0.219	22	0.279	0.857	19	0.219	25	0.286	0.857	19	0.219	22	0.281	0.857
8	4	15	0.148	12	0.078	0.571	17	0.162	15	0.074	0.643	15.4	0.15	11	0.078	0.586
12	4	13	0.219	23	0.280	0.857	13	0.219	20	0.245	0.857	13	0.219	21	0.267	0.857
9	5	21	0.200	28	0.149	0.786	23	0.219	25	0.286	0.857	21.4	0.204	24	0.168	0.800
13	5	17	0.238	38	0.485	0.929	19	0.219	34	0.221	0.857	17.4	0.234	31	0.416	0.914
14	6	23	0.238	35	0.479	0.929	25	0.238	35	0.477	0.929	23.4	0.238	34	0.478	0.929

In Table 2, the results are sorted in ascending order based on the distance from the I/O point. All three of the policies studied demonstrate similar relationships between the location stability measure and the retrievals to steady state. Although the different policies result in different locations approaching steady state more quickly or slowly, the longer times to steady state are always found in the locations with higher stability measures. The time it takes for each unit that originated in a location to approach steady state can be predicted directly from measures of the selective offloading policy (i.e., by examining the one-step transition matrix). Such knowledge can aid in assigning units with different mission criticality to storage locations and evaluating policies prior to operation. After 10 retrievals all items have less than a 0.50 probability of being located in their original locations.

The results from all seven configurations achieve similar performance. Locations closer to the I/O point tend to have (1) lower probability of retaining the same unit with the next retrieval, $p_{ll}^{(1)}$, (2) lower location stability, σ_l, (3) lower time to approach steady state behavior, RSS_l, (4) lower probability of remaining in the original storage location, U_{tl_0}, and (5) lower movement effort, E_l. If the probability that a location will retain its current unit and that location's stability measure are low, then the unit originating there will approach steady state more quickly. The knowledge of its location is lost more quickly. To further illustrate, Figure 2 is a heat map superimposed over the 7-by-5 configuration. For each location l, RSS_l is displayed and the locations approaching steady state more quickly are displayed in a darker color indicating a more negative result.

61	46	77	90	92
46	32	46	69	97
22	19	29	41	46
18	13	19	25	37
13	11	13	24	37
10	9	13	20	34
I/O	9	10	16	28

Figure 2: A heat map of the 7x5 configuation, where RSS_l for each location is displayed.

Units that originated in locations closer to the I/O point approach steady state behavior more quickly than units originating in locations farther from the I/O. This occurs because, in each of the policies, the units initially stored closest to the I/O point are the units that need to be moved out of the way to gain access to more deeply stored units. Given the locations closer to the I/O point, also require the lowest retrieval effort, a trade-off exists between retrieval effort and uncertainty management.

Insight 3: While locations closer to the I/O point in dense storage environments have lower retrieval distances, these locations will be more greatly impacted by unit location uncertainty.

When unit location information is assumed known, the existing storage literature, (e.g. Gu, Goetschalckx et al., 2007), reduces retrieval efforts through product allocation decisions assigning units requested more often to locations closer to the I/O point. Thus, this work advocates for considering the impact of unit location uncertainty when deciding product allocations in very high dense storage environments.

5.3 Impact of Multiple Communication Classes

Our models are used to examine a dense storage system with two storage classes. Storage class A and B both have nine communicating storage locations arranged in the configuration shown in Figure 3.

5a	6a	7a	8a	9a
	1a	2a	3a	4a
	1b	2b	3b	4b
5b	6b	7b	8b	9b

Figure 3. Dense Storage System with two communication classes both consisting of a 2 by 5 storage configuration.

The probability that a retrieval request is for units in storage class A is 75%, and for units in storage class B is 25% (i.e., $\Delta_A = 0.75$ and $\Delta_B = 0.25$). The puzzle policy is used for retrieval operations. The location uncertainty metrics are presented in Table 3, sorted ascendingly based on the location's distance from the I/O point. The performance of each class is benchmarked against a system equivalent with nine locations, all in a single communication class.

Table 3: Location performance measures for two, 2-by-5 storage classes and differing demand

			Single Communication Class				Communication Class A ($\Delta_A = 0.75$)				Communication Class B ($\Delta_B = 0.25$)			
L	I/O_d	E_l	σ_l	RSS_l	U_{10l_0}	$p_{ll}^{(1)}$	σ_l	RSS_l	U_{10l_0}	$p_{ll}^{(1)}$	σ_l	RSS_l	U_{10l_0}	$p_{ll}^{(1)}$
1	1	1	0.148	10	0.111	0.222	0.152	14	0.109	0.417	0.248	40	0.165	0.806
5	1	1	0.21	11	0.111	0.556	0.222	15	0.114	0.667	0.277	45	0.326	0.889
2	2	7	0.128	6	0.107	0.333	0.157	8	0.109	0.5	0.257	22	0.258	0.833
6	2	5	0.174	9	0.111	0.556	0.204	12	0.114	0.667	0.276	35	0.326	0.889
3	3	13	0.174	6	0.104	0.556	0.204	8	0.122	0.667	0.276	25	0.367	0.889
7	3	9	0.128	6	0.108	0.333	0.157	8	0.111	0.5	0.257	22	0.267	0.833
4	4	19	0.24	12	0.181	0.778	0.258	16	0.246	0.833	0.295	50	0.587	0.944
8	4	15	0.174	6	0.104	0.556	0.204	8	0.122	0.667	0.276	25	0.367	0.889
9	5	21	0.24	12	0.181	0.778	0.258	16	0.246	0.833	0.295	50	0.587	0.944

The locations in class B (which receives one quarter of the retrieval requests), take, on average, approximately three times longer to reach steady state. Thus, uncertainty in unit location diminishes more quickly in communication classes with higher demand probabilities. This is consistent with results empirically captured in retail operations, where inaccuracy was positively associated with demand (DeHoratius and Raman, 2008).

Insight 4: While all communicating classes will lose knowledge, communication classes with higher demand probabilities will lose knowledge faster than communication classes with lower demand probabilities.

5.4 Evaluation of Layouts

We use our model to evaluate the performance of different storage layouts. Existing literature has illustrated the inverse relationship between storage density and retrieval efforts; specifically, the storage density of any *c*-deep layout will be less than or equal to $2c/(2c+1)$ (Gue 2006). We are interested in understanding the relationship between storage density and location uncertainty. We consider four layouts displayed in Table 4: (1) an aisle-based system, (2) an inverse-T layout with accessibility constant *c* (Gue 2006), (3) a block layout of dimension *b* by *(b-1)*, and (4) a single escort puzzle layout of dimension *b* by *b*. Pallets are retrieved to the I/O point located midway up on the left-hand side of each layout. After the unit is retrieved to the I/O point, cases are selected and the pallet is put back based on the selective offloading policy before another unit is retrieved.

An aisle-based layout is common in retail distribution environments where minimizing retrieval effort is prioritized over maximizing storage density. An aisle-based layout is often denoted as a single-deep storage system (i.e., $c = 1$), because all units are directly accessible from an aisle. Thus, units are only moved when the unit is requested for retrieval. In our model, each location belongs to its own communication class (i.e., $|J| = n$), and the steady-steady performance for an aisle-based system is $\pi_{j\hat{\imath}} = \frac{1}{1} = 1 \ \forall \ \hat{\imath} \in C_j, \forall \ j \in J$. No loss of unit uncertainty occurs. The maximum storage density in an aisle-based system is 2/3.

In an inverse-T layout with accessibility constant *c,* at most *c-1* units are required to be moved out of the way to access a requested unit (Gue, 2006). In an inverse-T layout, the aisles form a spanning tree and units moved out of the way to gain access to others can be repositioned in aisle locations that do not block the path to the I/O point. For example, storage locations below the horizontal aisle can use the open aisle locations of the vertical aisle; while the storage locations above the horizontal aisle can use the aisle locations to the far left as repositioning locations. A "column policy" can be used to retrieve units stored in locations below the horizontal aisle, and a "row policy" can be used for locations above the horizontal aisle. Locations below the horizontal aisle communicate only with locations in the same column (and there are *c* such locations). Similarly, locations above the horizontal aisle communicate only with the *c* locations in the same row on the same side of the vertical aisle. This results in $|J| = \frac{n}{c}$ separate communication classes, all consisting of a total of *c* locations. For an inverse-T layout with accessibility constant *c*, the steady state performance is $\pi_{j\hat{\imath}} = \frac{1}{c} \ \forall \ \hat{\imath} \in C_j, \forall \ j \in J$ and the maximum storage density is *2c/(2c+1)*.

In a block layout of dimension *b* by *(b-1)*, a row policy can be applied, such that each row of *(b-1)* locations will form separate communication classes. There will be $|J| = \frac{n}{b-1}$ separate communicating classes all of cardinality *(b-1)*. In a block layout with a row policy, the steady state performance will be $\pi_{j\hat{\imath}} = \frac{1}{b-1} \ \forall \ \hat{\imath} \in C_j, \forall \ j \in J$, and the maximum storage density that can be achieved is $b(b-1)/b^2$.

In a single escort puzzle layout, a selective offloading policy will require movements in both the horizontal and vertical directions, and all $b^2 -1$ locations will communicate (i.e., $|J| = 1$) . The steady state performance will be $\pi_{j\hat{\imath}} = \frac{1}{b^2-1} \ \forall \ \hat{\imath} \in C_j, \forall \ j \in J$. As *b* approaches infinity, the storage density will approach 1.

Table 4: Evaluation of different layouts based on storage density and the steady state probability that a unit will be found in its originating location.

Traditional Aisle-Based Layout	Inverse-T Layout with Accessibility Constant *c*	Block Layout of Dimension *b* by *(b-1)*	Puzzle-Based System of Dimension b^2 with a Single Escort

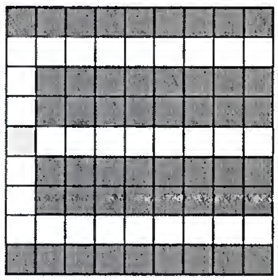
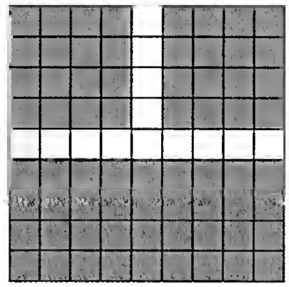
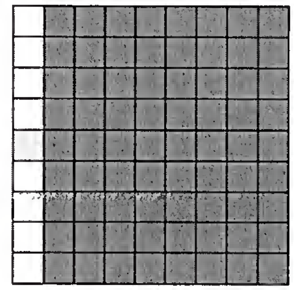
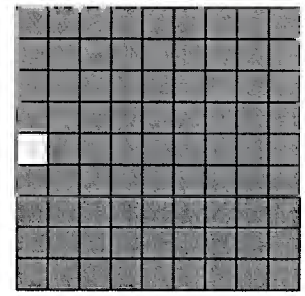
			
Storage Density Upper Bound = 2/3	Storage Density Upper Bound = *2c/(2c+1)*	Storage Density Upper Bound = $b(b\text{-}1)/b^2$	Storage Density Upper Bound = 1
Steady State Probability: *1/1 = 1*	Steady State Probability: *1/c*	Steady State Probability: *1/(b-1)*	Steady State Probability: $1/(b^2-1)$

Table 4 presents the results of the storage density and steady state uncertainty performance for the four layouts. The traditional aisle-based layout does not lose any location information, but has low storage density. Contrastingly, the puzzle-based system will approach 100% storage density, but retains a very low-level of location knowledge. These results illustrate,

Insight 5: There is an inverse relationship between storage density and steady state unit location probabilities.

This analysis highlights specific challenges associated with seabasing. Specifically,

Insight 6: Requiring selective offloading in dense storage operations leads to unit location uncertainty, which is not seen in traditional aisle-based storage operations.

5. Seabasing Case Study

A seabasing case study is provided to evaluate how best to move from a *Skin-to-Skin Replenishment* scenario to a *Tailored Resupply Packages* scenario. To do so, we utilize our uncertainty propagation model to quantity the trade-offs associated with different operational policies applied to the dense storage areas in a *T-AKE* ship. Operationally, a *Tailored Resupply Packages* scenario is more complex, requiring the ability to deliver emergent requests for tailored resupply packages by selectively offloading cargo stored in the cargo holds of ships (Parson, 2013; Pazour and Shin, 2016). The cargo requested will be for specific items at the unit or case level versus vessel-level demand requests of full pallet quantities common with a *Skin-to-Skin Replenishment* scenario. The primary performance objectives of a *Tailored Resupply Packages* scenario are to maximize responsiveness and to maximize storage density.

Data is obtained from load plans and inventory profile data for a dry cargo hold on the USNS Sacagawea. The USNS Sacagawea is a T-*AKE* ship targeted for use by the US military to fulfill a *Tailored Resupply Packages* scenario (Pazour and Shin, 2016; National Research Council, 2005); however, the load plan data is based on use for a *Skin-to-Skin Replenishment* scenario. Data are collected from a specific cargo hold containing a total of 475 dry cargo units, representing 47 different item-types. The data indicate a skewed inventory profile, where 17% of the item types, represented 80% of the total storage units. In our analysis, steady state is assumed and demand for items is proportional to the number of units in inventory, resulting in each unit being equally likely to be requested.

The expected search time for different operational policies is used as a surrogate for responsiveness. We assume human operators can move between locations while searching without moving units out of the way and convert the expected locations to search, defined in (18), to an expected search time. Let T_h denote the average time to move to the next storage unit stack horizontally, T_u the average time to search a storage unit, and H the average number of storage units in each stack of locations. The expected time to search before the unit originating in location l is found is denoted as $E(ST_l)$:

$$E(ST_l) = \frac{T_h}{H} E(SL_l) + T_u E(SL_l) = \left(\frac{T_h}{H} + T_u\right) E(SL_l) \tag{19}$$

Note, equation (19) relies on using our uncertainty propagation model to calculate $E(SL_l)$ the *expected locations to search* in steady state to identify the unit originating in location l. An overall search time in the hold is obtained by using a weighted average based on each item-type's inventory profile data.

Currently, the units are tracked in an inventory management system only at the cargo hold level and not at the individual unit location level. We assume T_h= 10 seconds, T_u = 30 seconds, and H= 2 units/stack. Thus current-state expected search time performance is calculated for the hold using equation (19) to calculate the expected search time for each of the 47 different item types. The current-state operational policy results in an expected search time of 18.6 minutes. This analytical result is supported by time study data obtained from a military exercise observing the ability to find and retrieve items in a dry cargo hold on a T-AKE ship (Sullivan, 2012). While this operational policy does not perform well in terms of responsiveness, such a system has high storage density potential and does not require enforcement of storage or retrieval policies.

Next, we compare the current state to an operational policy where items are segregated into subsections. The load plan data indicate some items are physically segregated from other items based on natural barriers, resulting in four communicating classes consisting of class 1 with 160 units (2 items), class 2 with 168 units (3 items), class 3 with 73 units (5 items), and class 4 with 74 units (37 items). Reapplying our uncertainty propagation model, results in reduced $E(SL_l)$ for the items. This results in an expected hold search time of 3.48 minutes; however, the expected search time in class four is 13.20 minutes. This policy improves greatly upon the responsiveness of the current-state operations, but requires items to be stored in a designated subsection of the hold, and requires more granularity of information to be recorded in the inventory management system than currently is being recorded.

Finally, we analyze an operational policy enforcing both storage and retrieval policies. Specifically, we enforce a column retrieval policy for class 4. Class 4 consists of 74 units, stacked 2 high in 6 columns of 12 units and 1 column of 2 units. Enforcing a column retrieval policy, results in an expected search time of 2.35 minutes. This is an 82% improvement in terms of responsiveness over the segregated policy, but requires enforcing both storage and retrieval policies. Also, access to each of the columns is required, which reduces the storage density potential.

This section illustrates, through a case study, how a decision maker interested in evaluating a number of operational policies for dense storage environments could use our developed model and metrics to quantify the responsiveness objective. Other operational policies can be explored using our models; for example, the trade-offs associated with stacking items higher to make room for more aisles and access points, as well as exploring the impact of different layouts or put back strategies.

6. Conclusions and Future Research

A finite-state discrete time Markov chain model is developed for a dense storage environment requiring retrieval of specific units. This work describes mathematically how unit location uncertainty originates in such a system, quantifies that uncertainty, and demonstrates how it spreads throughout the system given different operational policies governing how units are stored and rearranged during retrievals. For any non-depleting, communicating set of storage locations, the steady state location probability distribution is uniform across communicating locations.

As no known work has addressed the possibility of changing unit location uncertainty in warehouse operations, this work's focus on location uncertainty adds a new dimension to warehousing literature. The model, metrics, and case study presented in this paper are used to provide insights into the operation of very high density storage systems with selective offloading capabilities. These include the impact of a retrieval, training, and layout policies on the defined uncertainty metrics and existing retrieval metrics. The impact of uncertainty in dense storage systems can be mitigated; for example, the selective offloading policy and layout do affect the propagation of unit location uncertainty. However, performance trade-offs among the impact of uncertainty and two commonly used metrics in the literature: retrieval time and storage density, exist. Specifically, optimality in terms of minimizing expected movement effort assuming known unit location information does not imply optimality in terms of uncertainty management. Also, easier to retrieve from locations tend to have higher uncertainty performance than locations that are more difficult to retrieve from. An inverse relationship exists between storage density and steady state unit location probabilities.

Future work could include a depleting system model to incorporate storage operations in addition to retrievals and would need to account for varying numbers of storage units and a changing layout over time. The model could be expanded to include non-uniform demand distributions within communication classes and to consider non-stationary retrieval policies that are adapted strategically in response to the current state of the system. Finally, while the goal of this research is to develop descriptive models, an extension is to develop Markov decision processes or other prescriptive models for operations, as well as prevention and corrective policies.

Acknowledgements

This research was supported by the Office of Naval Research Young Investigator grant number N000141310594 and N000141612870.

Appendix A: Proof for the Expected Number of Locations to Search

Given the storage system has reached steady state, the expected number of locations to search for the unit that originated from location $l \in C_j$ containing one of the k_l copies is defined as $E(SL_l)$ and is derived here. From the definition of expected value, $E(SL_l)$ is defined as

$$E(SL_l) = \sum_{l'=1}^{|C_j|-k_l+1} l' \frac{\binom{|C_j|-l'}{k_l-1}}{\binom{|C_j|}{k_l}} = \frac{1}{\binom{|C_j|}{k_l}} \sum_{l'=1}^{|C_j|-k_l+1} l' \binom{|C_j|-l'}{k_l-1} \ \forall l \in C_j \ \forall j \in J \quad (20)$$

Then defining the numerator as Q.

$$Q = \sum_{l'=1}^{|C_j|-k_l+1} l' \binom{|C_j|-l'}{k_l-1} \quad (21)$$

From Pascal's Formula, the relationship below holds.

$$\binom{|C_j|-1}{k_l-1} = \binom{|C_j|}{k_l} - \binom{|C_j|-1}{k_l} \quad (22)$$

Thus, Q simplifies to

$$Q = \binom{|C_j|}{k_l} + \binom{|C_j|-1}{k_l} + \binom{|C_j|-2}{k_l} + \cdots + \binom{k_l+1}{k_l} + \binom{k_l}{k_l} \quad (23)$$

Substituting the relationship provided in Pascal's formula for each term in Q, results in the final reduced expression for Q,

$$Q = \binom{|C_j|+1}{k_l+1} - \binom{|C_j|}{k_l+1} + \binom{|C_j|}{k_l+1} - \binom{|C_j|-1}{k_l+1} + \cdots + \binom{k_l+1}{k_l+1} + \binom{k_l}{k_l+1} = \binom{|C_j|+1}{k_l+1} \quad (24)$$

Substituting the reduced expression for Q into the expression for $E(SL_l)$ completes the proof.

$$E(SL_l) = \frac{\binom{|C_j|+1}{k_l+1}}{\binom{|C_j|}{k_l}} = \frac{|C_j|+1}{k_l+1} \ \forall l \in C_j, \forall j \in J \quad (25)$$

End Proof.

REFERENCES

Allen, D. L. (2006). *Is Sea Based Sustainment Achievable by 2015*? Masters Thesis, U.S. Army War College; available at: http://www.dtic.mil/dtic/tr/fulltext/u2/a449447.pdf

Alpern, S. (2012). Rendezvous search games, Wiley Online Library.

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

Camdereli, A. Z., and J. M. Swaminathan. (2010). Misplaced Inventory and Radio-Frequency Identification (RFID) Technology: Information and Coordination. *Production and Operations Management*, 19 (1), 1-18.

Carlo, H. J., I. F. Vis, and K. J. Roodbergen. (2014). Storage yard operations in container terminals: Literature overview, trends, and research directions. *European Journal of Operational Research*, 235 (2), 412-430.

Chen, L., and A. J. Mersereau. (2015). Analytics for operational visibility in the retail store: The cases of censored demand and inventory record inaccuracy. Retail Supply Chain Management, Springer, 79-112.

Chick, S. (2012). An Introduction to Joint Operations on and from the Sea. Combined Joint Operations from the Seas Centre of Excellence, accessed August 31, 2015 at: http://www.cjoscoe.org/images/12_022-02_Encl1_CJOS_Handbook_Joint_Ops_from_the_Sea.pdf

Chung, T. H. and J. W. Burdick (2012). "Analysis of search decision making using probabilistic search strategies." IEEE Transactions on Robotics, **28**(1): 132-144.

Chuang, H. H. C., and R. Oliva (2015). "Inventory record inaccuracy: Causes and labor effects. *Journal of Operations Management*, *39*, 63-78.

Craig, B. A., and P. P. Sendi (2002). "Estimation of the transition matrix of a discrete-time Markov chain." Health Economics **11**(1): 33-42.

DeHoratius, N., A. J. Mersereau, A. J., and L. Schrage (2008). Retail inventory management when records are inaccurate. *Manufacturing & Service Operations Management*, *10*(2), 257-277.

DeHoratius, N., and A. Raman (2008). Inventory record inaccuracy: an empirical analysis. *Management Science*, *54*(4), 627-641.

Derhami, S., J. S. Smith, and K. R. Gue (2016). Optimising space utilisation in block stacking warehouses. *International Journal of Production Research*, 1-17.

Ergen, E., B. Akinci, and R. Sacks (2007). Tracking and locating components in a precast storage yard utilizing radio frequency identification technology and GPS. *Automation in Construction*, 16 (3), 354-367.

Futcher, F. W. (2003). Selective Offload Capability Simulation (SOCS): An Analysis of High Density Storage Configurations. Masters Thesis, Naval Postgraduate School.

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

Gu, J., M. Goetschalckx, and L. F. McGinnis (2010). Research on warehouse design and performance evaluation: A comprehensive review. *European Journal of Operational Research*, 203 (3), 539-549.

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., K. Furmans, Z. Seibold, and O. Uludag. (2014). GridStore: a puzzle-based storage system with decentralized control. *IEEE Transactions on Automation Science and Engineering*, 11 (2), 429-438.

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

Jang, D. W., S.W. Kim, and K. H. Kim. (2013). "The optimization of mixed block stacking requiring relocations." *International Journal of Production Economics* 143.2: 256-262.

Kadane, J. B. (2015). "Optimal discrete search with technological choice." Mathematical Methods of Operations Research: 1-20.

Kang, Y., and S.B. Gershwin (2005). Information inaccuracy in inventory systems: stock loss and stockout. *IIE Transactions*, *37*(9), 843-859.

Kang, K., and K. R. Gue. (1997). Sea based logistics: distribution problems for future global contingencies. Proceedings of the 29th conference on Winter simulation, IEEE Computer Society.

Karlin, S. (2014). A first course in stochastic processes, Academic press.

Kemp, J. (2008). *Viewing the Future of Seabasing Through the Lens of History: A Historical Analysis of Seabasing and What It Says About the Concept's Future Applicability*. Masters Thesis. US Marine Corps Command and Staff College. Available at: http://www.dtic.mil/dtic/tr/fulltext/u2/a491160.pdf

Kök, A. G., and K. H. Shang. (2014). "Evaluation of cycle-count policies for supply chains with inventory inaccuracy and implications on RFID investments." *European Journal of Operational Research* 237 (1), 91-105.

Larsen, C., and M. Turkensteen. (2014). "A vendor managed inventory model using continuous approximations for route length estimates and Markov chain modeling for cost estimates." *International Journal of Production Economics* 157: 120-132.

National Academies of Science, (2014). Force Multiplying Technologies for Logistics Support to Military Operations. National Research Council of the National Academies.

National Research Council (2005). Sea Basing: Ensuring Joint Force Access from the Sea, Committee on Sea Basing, DOI: 10.17226/11370, available at: http://www.nap.edu/catalog/11370/sea-basing-ensuring-joint-force-access-from-the-sea

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

Pazour, J. A. and I. Shin (2016), Logistics Models to Support Order-Fulfillment from the Sea. Progress in Material Handling Research: 2016, Material Handling Institute, Charlotte, NC.

Puterman, M. L. (2014). Markov decision processes: Discrete stochastic dynamic programming. John Wiley & Sons.

Reilly, P. (2015). Propagation of Unit Location Uncertainty in Dense Storage Environments Masters Thesis, University of Central Florida.

Ross, S. (1996). Stochastic Processes, 2nd Ed. John Wiley, 163-185.

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

Scala, N, and J. Pazour (2016). A Value Model for Asset Tracking Technology to Support Naval Sea-based Resupply. *Engineering Management Journal*, 28 (2).

Shechter, S. M., F. Ghassemi, Y. Gocgun and M. L. Puterman (2015). "Technical Note—Trading Off Quick versus Slow Actions in Optimal Search." Operations Research **62**(2): 353-362.

Song, J., C. T. Haas, C. Caldas, E. Ergen, and B. Akinci. (2006). Automating the task of tracking the delivery and receipt of fabricated pipe spools in industrial projects. *Automation in Construction*, 15 (2), 166-177.

Sullivan, R. (2012). T-AKE Support for Small-Scale Operations Ashore; Observations from Coconut Grove 12 - Quicklook Report. United States Marine Corps. Quantico, Virginia.

Vasili, M., S. H. Tang, and M. Vasili. (2012). Automated storage and retrieval systems: A review on travel time models and control policies. Warehousing in the Global Supply Chain. Springer, 159-209.

Yopp, J. (2006). Future Seabasing Technology Analysis: Logistics Systems, DTIC Document.
Yu, Y., R. B. De Koster. 2012. Sequencing heuristics for storing and retrieving unit loads in 3D compact automated warehousing systems. *IIE Transactions*, 44 (2), 69-87.

Yu, Y., & M. B. M. De Koster (2009). Designing an optimal turnover-based storage rack for a 3D compact automated storage and retrieval system. *International Journal of Production Research*, *47*(6), 1551-1571.

Zais, M., & D. Zhang (2016). A Markov chain model of military personnel dynamics. *International Journal of Production Research*, *54*(6), 1863-1885.

Zhang, A. N., M. Goh, and F. Meng. (2011). Conceptual modelling for supply chain inventory visibility. *International Journal of Production Economics*, 133 (2), 578-585.

Chapter 3: A Value Model for Asset Tracking Technology to Support Naval Sea-based Resupply

This chapter has been published in the following peer-reviewed journal:

Scala, Natalie, and Pazour, Jennifer A., 2016, "A Value Model for Asset Tracking Technology to Support Naval Sea-based Resupply," *Engineering Management Journal* (Special Issue on Engineering Management in the Military), 28.2, 120-130.

A Value Model for Asset Tracking Technology to Support Naval Sea-based Resupply

A value model is developed for military logistics that fulfills emergent requests for tailored resupply packages from the sea. Asset tracking technologies, including Radio Frequency Identification (RFID), barcoding, Internal Positioning Systems (IPS), and camera-aided technology, are considered as alternatives to a multi-objective decision model. Model measures include registration of inventory in the system, stowage factor enablement, storage location precision, retrieval identification accuracy, system compatibility, and security. We present the decision model, using insights from subject matter experts. Given the requirements of selective offloading in dense storage environments, IPS is the preferred asset tracking technology. Sensitivity analysis and recommendations for engineering managers are provided.

Introduction

This article presents a decision model to support logistics associated with military scenarios that require fulfillment of personalized resupply packages from the sea. Naval sea-basing is the process of moving, storing, locating, and redeploying various cargo located on military vessels on open water. This research focuses on the internal cargo flow processes on vessels required to handle requests for tailored resupply packages. These requests are emergent, which means that the requests occur without warning and result in high levels of variability both in when the requests will be made and in what will be requested. To identify asset tracking technology alternatives that may increase efficiency and responsiveness of storage, location, and retrieval of inventory, a Value-Focused Thinking approach is taken. The alternatives considered are barcoding, radio frequency identification devices (RFID), internal positioning systems (IPS), camera-aided technology, or doing nothing. Each alternative is an asset tracking technology that a naval sea-basing system could use to assist in managing inventory. Our model will assess these technologies on their value and identify the preferred technology.

This article begins with a discussion of naval sea-basing logistics, Value-Focused Thinking, and multiple objective decision analysis. Next, the model developed in this work is presented. Finally, results, sensitivity analysis, and implementation recommendations for engineering managers are provided.

Sea-based Logistics

Sea-basing is a military concept where tactical support is provided from the sea, rather than from the shore. From a logistics perspective, sea-basing transforms a set of vessels into floating distribution centers that can provide vital cargo requested for a variety of missions. Sea-based logistics involves determining how best to receive, store, retrieve, and deliver cargo from a vessel at sea.

The cargo flow processes that occur on a vessel involved in fulfilling emergent requests for resupply packages can be divided into the five following functions: (1) the transfer of cargo between ships, (2) the strike-down process, (3) the storage process, (4) the strike-up process, and

(5) delivery of the items to their objective location. The strike-down process is the transfer of cargo from the ship onboard point to stowage spaces, and the strike-up process is the transfer of cargo from the stowage space to the offload point for transfer to a delivery vehicle (e.g., high-speed vessel or aerial delivery). Given our focus is on fulfilling personalized resupply packages, we focus on palletized and breakbulk dry cargo, which is typical inventory for a sea-base, during the strike-down, storage, and strike-up processes.

Effective space utilization is an important consideration in logistics systems and is especially important in sea-based military logistics. In such systems, the ability to selectively offload cargo in high space utilization environments is required. The dry cargo storage holds are currently operated manually with workers using forklifts or pallet jacks to store, retrieve, and relocate pallets and containers. Each ship generates a load plan that, ideally, is followed perfectly, giving initial certainty in unit locations. Once in operation, ships receive orders requesting specific units and quantities to be retrieved, which starts the strike-up process. This need to retrieve specific units, perhaps even ones located in inconveniently placed locations, is known as selective offloading and is analogous to the concept of order fulfillment in the warehousing literature (Pazour & Meller, 2011). Currently, assets are not automatically tracked using asset tracking technologies that provide (x, y, z) coordinates of asset location. Thus, any shifting in cargo location while retrieving or searching for items results in unit location uncertainty as the system operates. Knowledge of unit location in dense storage environments has been observed to be lost over time and has resulted in non-value added time spent searching for the requested unit. For example, item location uncertainty when searching for requested crates was observed in an exercise conducted in 2012, which had a goal of observing the physical capability of ships to handle emergent requests for tailored resupply packages (Sullivan, 2012).

A number of peer-reviewed studies have been conducted on sea-based logistics operations. Kang and Gue (1997) built a simulation model to evaluate the performance of in-stream offloading of containers. Gue (2003) developed an optimization model to determine the supply chain network design for distributing cargo to mobile supply units when sea-based distribution is incorporated with land-based distribution. Brown and Carlyle (2008) developed an integer linear program to minimize shortages and maximize utilization of transports and total volume delivered by a specified combat logistics force. Salmerón, Kline, and Densham (2011) produced a global fleet station mission planner tool that enables the user to explore the feasibility of a United States Navy humanitarian engagement plan. During use, the tool allows for the consideration of scenario-specific constraints while seeking to optimize the route and schedule that maximizes the total mission value earned. The focus of each of the aforementioned works has been on macro-level supply chain issues, where cargo is assumed to be on the flight deck (thus, ignoring internal cargo flow operations) and unit location information of cargo stored on the ships is known. More recently, two papers motivated by sea-based logistics have developed models of internal cargo flow processes with item location uncertainty. Reilly, Pazour, and Schneider (2015) provide a mathematical foundation to describe the observed behavior of unit location uncertainty in seabasing logistics environments, which require both dense storage and selective offloading capabilities. Awwad and Pazour (2015) study the problem of searching for a single item in dense storage systems with uncertainty of item locations using a single searcher.

The United States Joint Deployment and Distribution Enterprise, which is comprised in part by a number of defense agencies and combatant commands, has goals to achieve end-to-end asset visibility and in-transit visibility. Asset visibility provides the "capability to see and redirect strategic and operational flow in support of current and projected priorities" by enabling timely and accurate information on the location, movement, status, and identity of assets, while in-transit visibility is the ability to "track the identity, status, and location of units, nonunit cargo, passengers, patients, and personal property from origin to consignee or destination" (Joint Chiefs of Staff, 2010, p. 15).

There are specific challenges and requirements with achieving asset visibility for tactical-level decision making for distribution processes in a sea-basing environment. These include having limited storage and operational space capacity, requiring the ability to selectively offload cargo, needing increased security measures, and synchronization of sea-based logistics operations with land operations. Exhibit 1 illustrates these challenges and requirements, and how they affect the sea-basing environment, in an influence diagram.

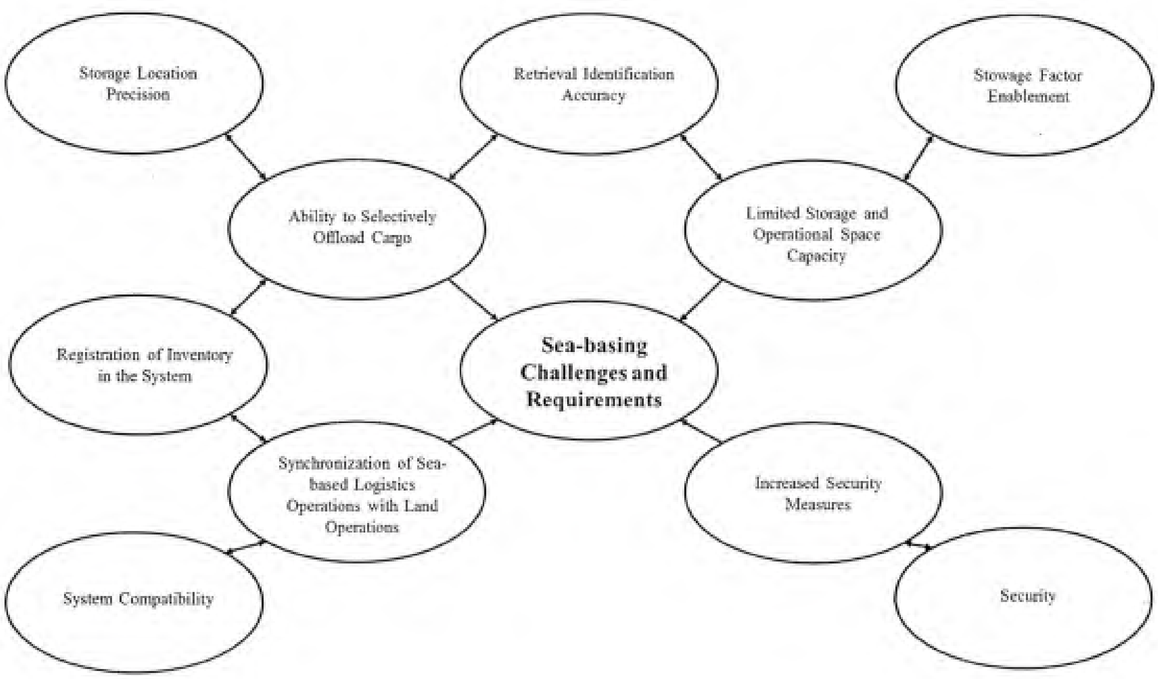


Exhibit 1: Influence diagram of challenges with asset tracking in the sea-base environment

Imperfect visibility associated with sea-basing can consist of a lack of knowledge of item location, item quantity, anticipated storage and retrieval schedules, as well as other factors. These types of imperfect information can impact system performance by decreasing throughput associated with storage and retrieval activities, increasing labor demands, or affecting the feasibility of meeting certain customer requirements. There has been research on the impact of imperfect visibility on logistic performance, but the focus has been on retail and not sea-based logistics (Buyurgan, Rossetti, & Walker, 2010; Kök & Shang, 2007). In this work, we create a decision model based on the challenges and requirements identified in our influence diagram. The model converts needs into measures that evaluate asset tracking technology devices that may increase efficiency and responsiveness for internal cargo flow processes. We also contribute to the understanding of logistics performance in the sea-based environment. We employ Value-Focused Thinking and a multi-objective model.

VFT and MODA

Value-Focused Thinking (VFT) is a decision analysis process that is designed to stimulate meaningful development of a decision model while supporting creative thinking about the problem at hand; its purpose is to enable clear and explicit definition of the decision problem (Keeney, 2008). The creative thinking component of VFT allows the decision maker to brainstorm all possible alternatives and objectives for the problem, from a value perspective. At times, improved alternatives are identified or previously unconsidered aspects of the problem are brought to light. This process leads to definition of a better set of objectives, identification of all alternatives, and a more holistic decision model. The decision maker acts proactively instead of reactively in the decision process, using value to define all that he or she cares about within the context of the problem. Further details on the VFT process can be found in Keeney (1992, 2008).

The United States Department of Defense has used the VFT process with multiple objective decision analysis (MODA) (Dillon-Merrill, Parnell, Buckshaw, Hensley, & Caswell, 2008). MODA is a decision analysis technique for evaluating a decision under multiple objectives or criteria, and the objectives may be conflicting (Parnell, 2007; Keeney & Raiffa, 1976; Kirkwood, 1997). Related methods to MODA include multiple attribute utility theory (MAUT) and multiple criteria decision analysis (MCDA). MODA is a utility approach, where values and preferences are employed to rank alternatives subject to limitations. In the MODA process, objectives are defined in terms of measurable metrics on which the alternatives are evaluated. The measures are organized into a value hierarchy, which has a similar form and function to decision hierarchies created for simple multi-attribute rating technique (SMART) and the Analytic Hierarchy Process (AHP). Then, for each measure on the MODA hierarchy, a value function is defined. These functions can be continuous or discrete and identify both the worst outcome (score of 0) and the best outcome (score of 100). Moderate outcomes in between best and worst are also defined. Alternatives are then scored by creating an additive value function across the value objectives and measures for each alternative (Keeney & Raiffa, 1976). The alternative with the highest score is deemed preferred. The robustness of the decision is examined through sensitivity analysis.

MODA and VFT rely heavily on the inputs of a subject matter expert (SME), who brainstorms measures and alternatives, defines the value functions, and develops weights for every measure. This elicitation process is typically facilitated by a decision analyst. Thus, appropriate and knowledgeable SMEs with significant contextual background in the problem area should be chosen for the model building process.

MODA allows for the addition of alternatives during the model building process. As SMEs think critically, additional value-added activities may be identified. In fact, SMEs are encouraged to initially identify as many alternatives as possible, as part of the VFT process. Additional alternatives can simply be scored and added to the analysis. This approach provides an advantage over other decision modeling techniques, such as AHP. Unlike AHP, alternatives with MODA can be added without redoing pairwise comparisons. For example, the value functions associated with measures may change over time or a new alternative may need to be added for consideration. Using VFT and MODA allows us to simply update a value function as necessary or create a new value function if needed. With the AHP, pairwise comparisons would have to be redone; as the definition of a measure would change, so would the corresponding assessments of the decision maker. This is tedious and impractical. With VFT, weights for the measures can be updated, alternatives can be rescored on only the revised or additional measures, and the ranking of alternatives would be updated. Rank reversal is not a potential issue. Furthermore, using VFT promotes brainstorming on problem objectives and subsequent creation of alternatives to help satisfy the objectives, whereas methods such as AHP focus on identification of alternatives first and then attributes to evaluate those alternatives (Goodwin & Wright, 2004). As a result, VFT allows for inclusion of alternatives that may not have been originally considered and allows focus on fundamental values of the decision problem.

MODA can handle a complex set of measures to evaluate the objectives. Most decision problems have at least three to five measures, while very complex problems can have up to 100 measures (Scala, Kutzner, Buede, Ciminera, & Bridges, 2012). Further details on the MODA process can be found in Parnell (2007), Keeney & Raiffa (1976), and Kirkwood (1997). A step-by-step outline of the process can be found in Dillon-Merrill et al. (2008).

MODA and VFT have been used in many United States defense-related applications, including base realignment and closure (BRAC) (Ewing, Tarantino, & Parnell, 2006), system components for air and space dominance (Parnell, Conley, Jackson, Lehmkuhl, & Andrew, 1998), psychological operations (Kerchner, Deckro, & Kloeber, 2001), nuclear terrorism protection (Feng & Keller, 2006), workforce planning (Scala et al., 2012), energy transformation (Simon, Regnier, & Whitney, 2014), and Air Force cyber investment (Parnell, Butler, Wichmann, Tedeschi, & Merritt, 2015). The method has also been used in applications not related to defense; an example is transportation disruption response (Tong, Nachtmann, & Pohl, 2015). Value creation models are also used in portfolio analysis; an example is Kirchhoff, Merges, & Morabito (2001). A review of additional military applications using VFT can be found in Keefer, Kirkwood, and Corner (2004). A full survey of VFT applications can be found in Parnell, et al. (2013). Dillon-Merrill, et al. (2008) identify pitfalls and best practices in Department of Defense related models; decision analysts and engineering managers are urged to consider these when utilizing MODA approach.

Model for Asset Tracking Technologies

Our model employs the combined standard, as defined by Parnell, Bresnick, Tani, and Johnson (2013), utilizing a mix of key policy documents and interviews with SMEs. The combined standard of model development is very common in decision-making and incorporates elements of both the platinum (SME driven) and gold (document driven) standards. In this work, we had access to two SMEs provided by the United States Office of Naval Research. The first SME is the president of a consulting firm, has completed several sea-based logistics studies, and is a NATO Civil Expert for sea-basing operations via the Maritime Administration. The second SME has also completed several sea-based logistics studies, including asset tracking projects to locate containers in a depot using asset tracking technologies. The SMEs have similar backgrounds and comparable levels of extensive expertise related to both naval and sea-basing applications. They also have global expertise in asset tracking, outside of the naval and sea-base application areas. Unal, Keating, Chytka, and Conway (2005) recommend such varied expertise in order to reduce bias in SME input. Development of the value hierarchy began with documentation that has either been created or approved by senior decision makers within the U.S. Navy and U.S. Government in order to identify potential measures (e.g., Clark, 2002; Congressional Budget Office, 2007; Department of the Army, 2008; Gunderson, Canfield, Dann, & McCambridge, 2004; Mallon, 2008; Moore & Hanlon, 2003; National Research Council Committee on Sea Basing, 2005; Naval Research Advisory Committee, 2004). These potential measures were reviewed with our SMEs, who refined the list and created definitions for every measure. The value functions were directly elicited from the SMEs in a series of virtual meetings, which both SMEs attended. The meetings were virtual because the SMEs were geographically dispersed. From this iterative process, a hierarchy was developed from the six final measures discussed in the next section, with each measure on the same level. This hierarchy is both mutually exclusive and collectively exhaustive, a key tenant of a MODA (Parnell, et al., 2013).

A MODA model is comprised of the measures organized in a value hierarchy, definition of value functions for each measure, calculation of weights for every measure, scoring of the alternatives on each measure, and sensitivity analysis of the results. The next section presents the details of the components of the model.

Measures and Value Functions

Through a series of iterative meetings, a set of measures used to evaluate asset tracking alternatives was confirmed by the SMEs. To arrive at this set of measures, taking a process view of the internal cargo flow processes required to fulfill emergent requests in a sea-basing environment assisted in identifying the main concerns in the decision-making situation. For each function, the challenges and requirements associated with sea-basing decision making were identified and grouped. This was done through a gold standard review and discussions with Navy and Marine Corps personnel. Next, the SMEs received a summary document that described the objective and scope of this study and included a gold standard list of measures. The SMEs were asked to evaluate the importance of the measures provided and were encouraged to include any additional relevant measures so that the list of measures was exhaustive. Through this iterative process, some gold standard measures were removed (e.g., maintenance and upkeep), because the SMEs determined them as not important to the sea-basing asset tracking decision, and others were added (e.g., system compatibility). Care was taken to ensure that all measures were important and were defined such that they were quantifiable. As illustrated in the influence diagram in Exhibit 1, each measure is quantifiable and addresses a sea-basing challenge or requirement. Also, each measure addresses significant and measurable areas that provide insight on the preferred asset tracking technology, given the considered alternatives.

The final measures are as follows:

1. *Registration of inventory in the system* is the time needed to setup or register receipt of the items on the ship for each alternative. This is measured per unit or per pallet. Once inventory is registered that it has arrived on the ship, the strike-down process begins.
2. *Stowage factor enablement* is the maximum stowage factor at which the tracking technology enables functional operations of selective offloading capabilities. Specifically, we consider the stowage factor at which the asset tracking technology enables reasonable packing of the holds. This is measured as percent of cube, defined as the cubic feet of stored cargo divided by the cubic feet of available space for storage.
3. *Storage location precision* is the granularity with which the item's storage location can be accurately marked or captured. This is measured by the precision level that is captured by the alternative tracking technologies and is recorded at the strike-down process. One way to measure this is using the granularity of information provided in the physical location codes, such as the deployable unit location number, which consists of nine characters of granular information (Department of the Navy, 1994). The first character represents the facility location, and the ninth character identifies the subdivision within the segment, such as a drawer or compartment level. Not all items receive a full nine-digit number, depending on the level of detail recorded at the strike-down process.
4. *Retrieval identification accuracy* is the measure of the difference between (1) where the requested item is recorded to be and (2) where it is actually found on the ship. Specifically, confirming that a location is accurate and the item is actually there, given that the item is marked to be in that location. The search process is initiated when a request for retrieval of an item arrives and is the beginning of the strike-up process. This is a mirror operation to *storage location precision* and is defined in terms of navigation to the item in question.
5. *System compatibility* is the ease with which the asset tracking technology can interface with existing Standard Military Information Systems (STAMIS) that are currently or envisioned to be used for In Transit Visibility Tracking and/or property accountability. This is an operational use and is measured by the ease of transferring data between the asset tracking and STAMIS systems.
6. *Security* is the potential for an adversary to intercept and/or hack signals emitted from the technology. Specifically, defining the ease or difficulty of detecting the produced signal. This is measured by the sophistication of the technology alternatives.

Value functions can be loosely defined as the marginality of the decision maker's preferences, and they do not consider risk attitudes (Goodwin & Wright, 2004). Value functions were elicited from the SMEs for each measure. SMEs were asked to identify the best and worst alternative performance for each measure, with the best performance receiving a value of 100 and the worst performance receiving a value of 0. SMEs were then asked to identify moderate levels of performance along with the corresponding value of each option. Exhibit 2 provides the value function definition for each measure; each value function is monotonic. As an illustrative example, Exhibit 3 graphs the value function for *registration of inventory in the system.*

Measure	Performance	Value
Registration of Inventory in the System	Instant	100
	0 to 1 minutes	95
	1 to 2 minutes	85
	2 to 3 minutes	50
	3 to 5 minutes	25
	Greater than 5 minutes	0
Stowage Factor Enablement	70% packed and above (dense)	100
	50% to 70% packed	40
	25% to 50% packed	20
	Up to 25% packed	0
Storage Location Precision	Item recorded to be in a specific subdivision/compartment within the level of the stack (ninth numerical position)	100
	Item recorded to be in a specific level within the stack (eighth numerical position)	95
	Item recorded to be in a specific stack of a specific aisle or row of a hold (sixth and seventh numerical positions)	90
	Item recorded to be in a specific aisle or row in a hold (fourth and fifth numerical positions)	80
	Item recorded to be in a specific hold on the ship (second and third numerical positions)	65
	Item recorded to be on the ship (first numerical position)	50
	Item cannot be recorded to be on the ship	0
Retrieval Identification Accuracy	Item recorded and found to be in a specific subdivision/compartment within the level of the stack (ninth numerical position)	100
	Item recorded and found to be in a specific level within the stack (eighth numerical position)	95

	Item recorded and found to be in a specific stack of a specific aisle or row of a hold (sixth and seventh numerical positions)	90
	Item recorded and found to be in a specific aisle or row in a hold (fourth and fifth numerical positions)	80
	Item recorded and found to be in a specific hold on the ship (second and third numerical positions)	65
	Item recorded and found to be on the ship (first numerical position)	50
	Item cannot be recorded to be on the ship	0
System Compatibility	Seamless between systems, real time updates	100
	Delayed process, but updating throughout the day	90
	Batch transfer that updates once per day	80
	Manual entry	50
	No compatibility	0
Security	Undetectable signal	100
	Detection of signal possible	85
	Detect signal and discern patterns	70
	Detect, read, and understand signal	35
	Actively enter and manipulate system	0

Exhibit 2: Value function definition for each measure

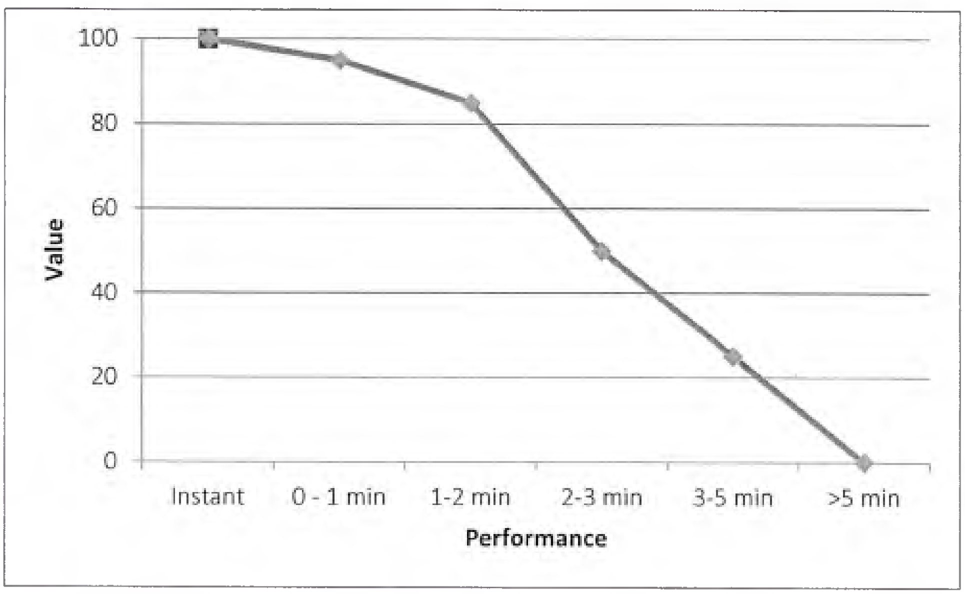


Exhibit 3: Value function for *registration of inventory in the system*

Weights

The rank order centroid method was used to develop weights for every measure. This method is a mathematical technique developed by Barron (1992) where the decision makers rank order the measures under consideration with the ranks converted to weights using the equation $w_k = \left(\frac{1}{K}\right)\sum_{i=k}^{K}\left(\frac{1}{i}\right)$. The weights sum to unity. The rank order centroid method is useful when multiple decision makers are involved, as achieving agreement on a direct weight can be difficult. The weights are derived systematically, by using implicit information in the ranks, and are not ad hoc, thereby outperforming other weighting methodologies (Barron & Barrett, 1996). The rank order centroid method minimizes the average utility loss, defined by the utility of the optimal strategy minus utility of a random strategy (Edwards & Barron, 1994) and has been used in other MODA applications (e.g. Scala et al., 2012).

The two SMEs provided ranks for the six measures. Both SMEs were present for the elicitation and were in agreement with the order of importance. Exhibit 4 presents the six measures in rank order along with their corresponding rank order centroid weight.

SME Rank	**Measure**	**ROC Weight**
1	Stowage Location Precision	0.4083
2	Stowage Factor Enablement	0.2417
3	Retrieval Identification Accuracy	0.1583
4	System Compatibility	0.1028
5	Registration of Inventory	0.0611
6	Security	0.0278
	Total Weight	1.0000

Exhibit 4: Rank order centroid weights for measures

Alternatives

For alternatives, we consider four asset tracking technologies for internal cargo flow processes within the sea-base environment: barcoding, radio-frequency identification devices (RFID), internal positioning services (IPS), and camera-aided technology. For comparison purposes, we also consider doing nothing as an alternative, implying there is no change to the current process.

Barcoding involves utilization of a numeric sticker to identify characteristics of an item. Barcoded scanning and product verification offers a logical and proven means to decrease errors in inventory (Oldland, Golightly, May, Barber, & Stolpman, 2015). A barcoding system requires scanning of the barcode on the item, as well as scanning the barcode associated with the storage location (also called a "license plate"). For an item to be assigned and mapped to a location with barcoding technology, each pallet location requires a license plate location. Thus, barcoding requires a single-deep storage system that could include items stacked on top of each other. Given this environment, an item can then be recorded to be in a specific level within the stack. Items could then be found in a specific stack of a specific aisle or row of a hold. While being stored, items may need to be reshuffled to gain access to other items stacked on top of each other. Given that location updating requires manual intervention, the *retrieval identification accuracy* measure can have a higher granularity than the *storage location precision* measure.

RFID utilizes a transponder with a reader to track items; thus, all items that require tracking are affixed with a transponder. The transponder's microchip passes data to the reader through an antenna, with the reader feeding the data to a computer for computations (Violino, 2005). Active RFID systems have transmitters that broadcast location, and material can be identified proactively by placing an RFID tag on a piece of inventory and locating that inventory in proximity to the reader. Passive RFID systems do not actively broadcast location, and material needs to be passed through a reader in order to be identified. In both RFID systems, data on an item's tag is sent to a reader. However, no positioning information is provided, only that a response has been sent and received. In this work, we consider RFID in a general sense and evaluate the alternative by using the main characteristics of an RFID system. A primary advantage of RFID over barcoding is that items do not have to be physically scanned or touched in order to be read.

An Indoor Positioning System (IPS) is an asset tracking technology that functions similarly to a Global Positioning System (GPS). However, IPS can be used in indoor spaces as it is ground based, whereas GPS is satellite based. An IPS system requires receivers to be attached to the items. In addition, multiple wireless transmitters are required such that item location coordinates can be triangulated. There are many ways to structure an IPS. However, the most common systems use Bluetooth or a wireless local area network (Liu, Darabi, Banerjee, & Liu, 2007; Kim, Seo, Krishna, & Kim, 2008).

A special case of an IPS is a wireless mesh network, where active wireless receiver nodes are capable of communicating with each other and of relaying information among the nodes (U.S. Patent No. 7,852,262, 2010). A wireless mesh network requires more than three nodes to triangulate item information. Wireless mesh networks can consume less power than a traditional IPS, increasing the system lifetime from a single power source. However, to consume less power, the distance between each transmitter must be small, which may require more transmitters. System cost and complexity increases as more transmitters are installed (Mao & Fidan, 2009). We consider IPS in a general sense and evaluate this alternative using the primary characteristics of an IPS system.

Camera-aided asset tracking technology is another potential alternative for sea-based logistics operations. Camera-aided technology can be implemented by affixing multiple digital cameras near the ceiling of each storage hold. Such a system requires that each cargo container has a label attached to it that is visible by the camera (e.g., a unique, large printed marking). Due to the dense storage requirement, items that are on the bottom of the stack are not able to be identified as those items are not in the camera's line of sight. Therefore, such a system provides partial asset location information. This setup also requires that the images from the cameras be sent to a computer, where either human or computer-aided machine vision is used to identify and map the location of each visible cargo container to locations in the storage hold.

In order to realistically bound the decision space, we make the following assumptions and observations.

1. The sea-base storage environment is dense and does not include racking.
2. RFID readers can be fixed to the doors of each hold, allowing inventory to be identified as it enters or exits the hold. As a result, RFID is not able to identify *(x, y, z)* coordinates of storage but is able to identify that an item has entered or exited the hold.
3. The maximum stowage factor in a single-deep storage system is 2/3 (Gue, 2006).
4. RFID tags and barcodes, which are mature technologies, have been placed on items by an upstream supply chain stage; items would be received on the ship affixed with these technologies.
5. Given that IPS is an emerging technology, the receivers required for IPS are assumed to be used only for internal cargo flow. These receivers, which can be a few inches thick, are required to be placed inside cargo when it arrives on the ship, such that stacking of cargo can still occur.
6. The camera-aided technology requires specialized markers that are visible by the camera. These markers, which can be fixed to the outside of the cargo, will be placed on the cargo when it arrives on the ship.

Results

Each alternative was scored on each of the six measures. To do so, a corresponding value was identified and assigned to an alternative for every measure, using policy documentation and technical specifications to support the gold standard of Parnell, et al. (2013). For example, based on the capabilities of the technology and model assumptions, IPS scored 25 for *registration of inventory in the system,* 100 for *stowage factor enablement,* 95 for *storage location precision,* 95 for *retrieval identification accuracy,* 100 for *system compatibility,* and 70 for *security*. The SMEs reviewed the gold standard scoring, providing feedback and validation; they were in agreement with the final scoring of all alternatives. Following standard MODA procedure, a weighted average score s_a for each alternative a was calculated using the following equation: $s_a = \sum_{i=1}^{m} w_i v_{ia}$ where w_i is the weight of measure i as defined in Exhibit 4, and v_{ia} is the value function scale item for measure i and alternative a as defined in Exhibit 2, with m total measures. In descending order, scores for every alternative are shown in Exhibit 5.

Alternative	Score
IPS	91.75
Barcoding	80.54
RFID	79.75
Camera	78.22
Do Nothing	60.42

Exhibit 5: Final scores s_a for all alternatives in descending order

Our results indicate that IPS are the preferred alternative, with barcoding scoring as the second best alternative. These results are logical, as there are many benefits to an IPS in a sea-based environment that requires selective offloading in dense storage environments. Most importantly, IPS is capable of identifying location coordinates of cargo in densely-packed storage systems and can have an accuracy of 1-5 meters with an 83% probability of finding a specific location within 1.5 meters (Liu et al., 2007). Also, IPS systems have the capability to process anywhere from 11-108 Mbps of data, which allows the entire system to update in seconds (Liu et al., 2007). Such characteristics result in IPS achieving the highest score among the alternatives for *stowage factor enablement, storage location precision*, and *retrieval identification accuracy*. A wireless receiver must be attached to the inside of each cargo container; thus, IPS requires more time to register inventory in the system than other alternatives. Also, given IPS uses a wireless system, it has the possibility of a detectable signal and is slightly less secure than other alternatives. However, the performance in other measures outweighs these factors.

Sensitivity Analysis

We conducted a sensitivity analysis for the results by varying the weight of each measure from 0 to 1, holding the other weights constant and normalizing the sum of all weights to 1. As described earlier, the weights were elicited via the rank order centroid method, which converted an ordinal raking to a ratio weight. Thus, the potential swing or change in a weight must be considered. Specifically, our sensitivity analysis examines if the preferred alternative changes due to a shift in weight. Weights could modify over time due to evolving priorities of the United States Navy as well as further development and use of the sea-based environment. Also, the ordinal ranking of the weights as elicited from the SMEs do not necessarily address their strength of preference of each measure.

IPS remains the preferred alternative for all weights of *storage location precision*, *retrieval identification accuracy*, and *system compatibility*. IPS remains the preferred alternative as long as the weight for *stowage factor enablement* is greater than about 0.05. For a weight to be that small, *stowage factor enablement* would have to drop to fifth or sixth in importance. This is highly unlikely due to the dense storage requirements of naval sea-basing, which place a premium on stowage factor. For *registration of inventory in the system*, IPS remains the preferred alternative as long as the weight for the measure is less than about 0.2. A great increase in the importance of that measure is unlikely. Also, our assumption is that IPS tags are required to be affixed to the inside of the cargo upon receipt, which results in a conservative estimate of the time required for inventory registration. As IPS technology matures, this time is likely to decrease. Finally, IPS remains the preferred alternative as long as the weight of the *security* measure remains below about 0.4. If such a swing in weight occurred, security would become the most important measure, minimizing the importance of the requirements and challenges of the dense storage environment. Clearly, defense scenarios exist when security is at a premium. However, the frequency of such would be ad-hoc, as sea-based logistics usually handles dry cargo, minimizing the amount of sensitive materials. Exhibits 6, 7, and 8 show the results from the sensitivity analysis for *stowage factor enablement, registration of inventory in the system*, and *security*. The x-axis represents the weight of the measure, with all other measure weights constant and normalized, while the y-axis represents the final score s_a for every alternative. Barcoding is primarily the alternative that becomes preferred as IPS loses dominance, but the likelihood that weights would swing enough to cause a change in alternative is slim. We conclude that the IPS alternative is the preferred choice for tracking naval sea-based inventory, given the challenges presented in Exhibit 1, and this recommendation is robust to changes in measure weights.

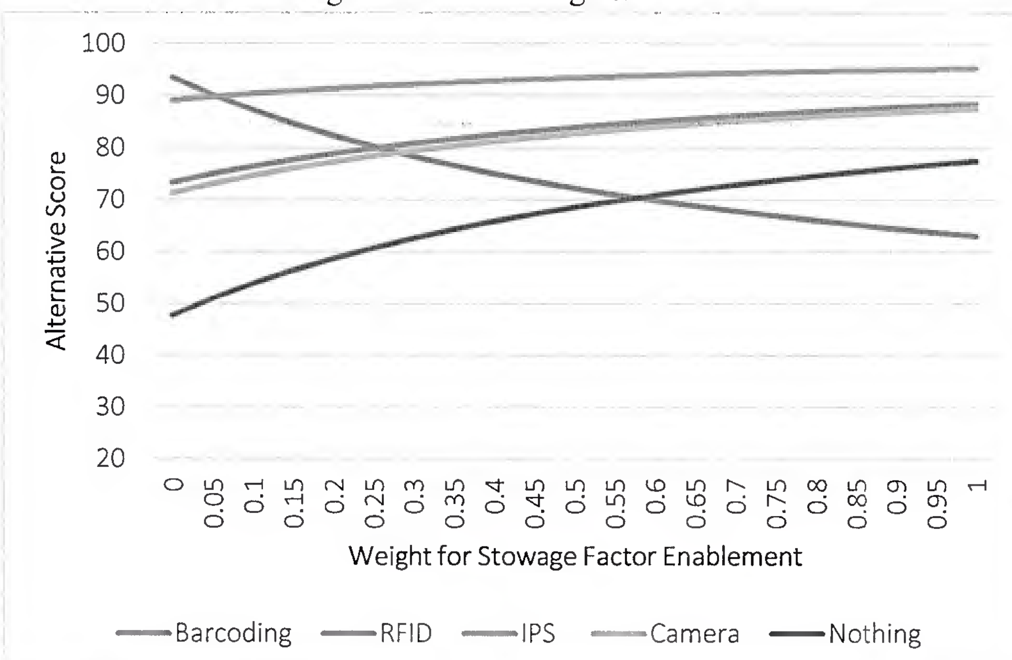


Exhibit 6: Sensitivity analysis for *stowage factor enablement*

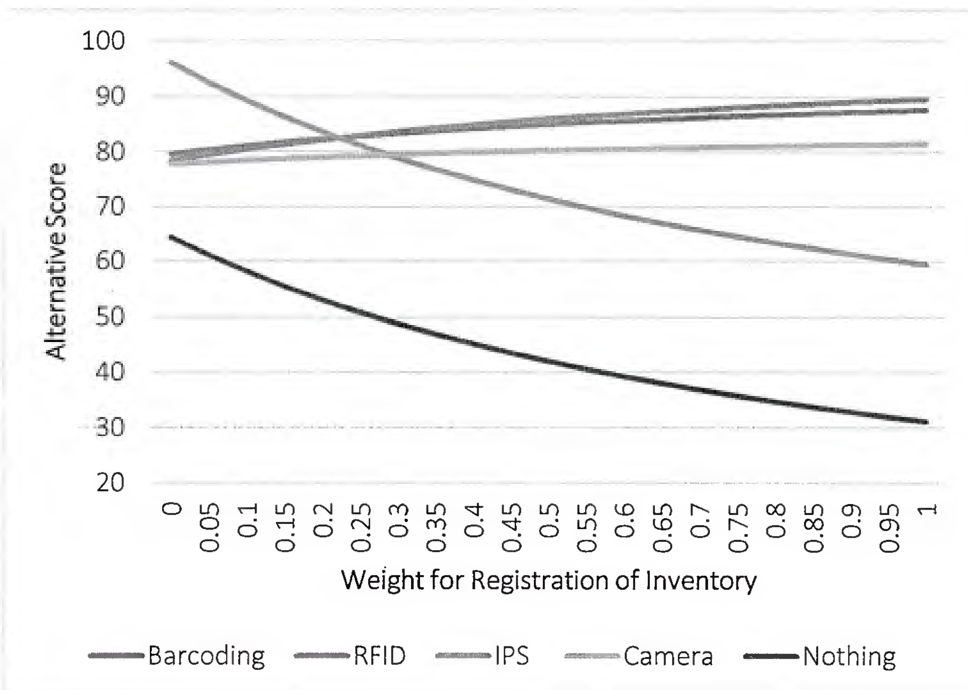


Exhibit 7: Sensitivity analysis for *registration of inventory in the system*

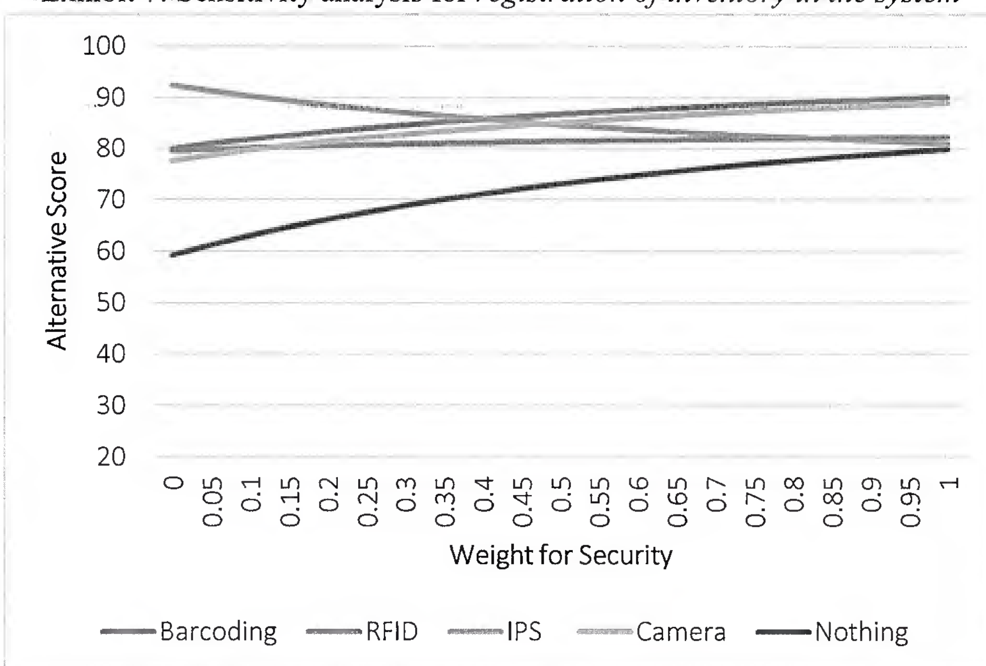


Exhibit 8: Sensitivity analysis for *security*

Validation and Implications for Engineering Managers

The SMEs reviewed the alternative scores and sensitivity analysis, concluding that the results were reasonable. Each alternative was initially scored using the gold standard, utilizing supporting United States Government and Navy documentation as well as industry specifications, to support the value assessment on every measure. The SMEs reviewed each value and provided feedback and appropriate modifications. Sensitivity analysis was also reviewed, with the SMEs confirming that drastic weight changes for *stowage factor enablement, registration of inventory in the system*, and *security* are unlikely to occur, supporting the robustness of the IPS alternative.

We reviewed our results with project sponsors at the Office of Naval Research and received positive feedback on both the approach taken and the results. Specifically, the recommendation for IPS resonated with the project sponsors, as the mission of fulfilling emergent requests in a sea-basing environment requires selectively offloading cargo in a dense storage environment. For other mission requirements, such as planned resupply of ships where the requests for cargo can be planned in advance and are typically for bulk requests (e.g. pallets of food or ammunition), less complex asset tracking technology may be preferred.

This study does not consider implementation cost as a tradeoff. The true cost to implement any alternative fleet wide would depend on a variety of factors including size of fleet, sourcing of inventory tags, and personnel training time. Availability of funds would be constrained by the congressional budget, and total cost of implementation would likely be driven by the contractor request for proposal process. Future work plans to estimate these costs and support the Office of Naval Research as needed in implementation of asset tracking technology in naval sea-basing.

A challenge to evaluation and acquisition of technologies is that information on newer technologies is difficult to obtain (Daim & Kocaoglu, 2008). IPS is an emerging technology and requires specialized transmitters and receivers. We have been conservative when evaluating this alternative, assigning values that could be lower than actual during implementation. As the technology matures, items could arrive with the tags pre-affixed, which would continue to shorten registration time and increase overall performance. Even with this guarded approach, IPS is still the preferred alternative.

When performing value-focused decision analyses, engineering managers should be careful to engage SMEs who have detailed knowledge of the application area and can reasonably understand the structure, purpose, and mechanics of a MODA model. The SMEs need not have a decision analysis background, but no knowledge of the MODA process may lead to elicitation of values that are unrealistic or inappropriate for the study.

A strong benefit of the VFT process is that assumptions are challenged and stakeholders arrive at a recommendation that is based on value, free of bias and preconceived notions. For example, at the start of this project, the initial reaction of our stakeholders was that RFID would be the preferred alternative. However, RFID scored rather poorly in the analysis and is clearly not the preferred alternative. RFID has limitations, primarily in that it does not provide coordinate location values; instead, RFID tags only identify the hold in which the item is located. Also, combining RFID with a hand-held reader that has coordinate capabilities is not feasible in a dense storage environment without reducing the storage factor enablement. Finally, in a sea-based environment, cargo is required to be densely stored in essentially a metal box structure, which can lead to issues with signals and battery life. Therefore, an important insight relevant to engineering managers is that RFID-tagged items alone will not identify cargo locations to the granularity required when dense storage and selective offloading are required. These requirements exist when sea-basing is used to fulfill tailored requests. An initial reaction without analysis may have advocated for RFID, but a full consideration of all important measures and associated performance proved otherwise. Thus, engineering managers who undertake VFT analyses should be open to the challenge of preconceived notions and be willing to coach the client into understanding and accepting the preferred alternative, even if it differs from original thought. A preferred alternative maximizes value and provides the client with a recommendation that aims to satisfy all needs.

Using VFT in defense applications is a unique process, as cost tends not to be considered, and defense posture or benefit is maximized. Engineering managers working in defense must be cognizant of the unique environment and focus their efforts on full elicitation of value and consideration of all possible alternatives.

VFT models have strong benefits and allow qualitative values to be quantified, supporting a data-driven recommendation. Such analyses can be extremely beneficial, and we encourage engineering managers to consider using this approach to evaluate alternatives in a multiple objective environment.

Conclusions

This article presents a value-focused multi-objective decision model to evaluate asset tracking devices for fulfilling emergent requests in a sea-basing environment. Model measures and value were elicited using SME input and policy documentation. Alternatives were scored, and IPS was found to be the preferred alternative for asset tracking when selectively offloading cargo in a dense storage environment is required. Engineering managers can employ Value-Focused Thinking models in practice utilizing the input of experienced SMEs, while remaining open to preconceived notions being challenged in the modeling process.

Acknowledgements

The authors would like to thank our two subject matter experts for their invaluable insights and time as well as Logan Minor, Vincent Schiavone, Eric Schiech, and Corinne Skala for their assistance with data collection. We sincerely appreciate their contributions to this project. This research is supported by the Office of Naval Research Young Investigator Program: The Design of Responsive Sea-Based Logistic Delivery Systems, Award Number N00014-13-1-0594 and N000141612870.

References

Awwad, M., & Pazour, J. (2015). *Search plan for a single item in an inverted T k-deep storage system.* Unpublished manuscript.

Barron, F. H. (1992). Selecting a best multiattribute alternative with partial information about attribute weights. *Acta Psychologica, 80,* 91-103.

Barron, F. H., & Barrett, B. E. (1996). Decision quality using ranked attribute weights. *Management Science, 42*(11), 1515-1523.

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

Buyurgan, N., Rossetti, M. D., & Walker, R.T. (2010). An analysis of imperfect RFID visibility in a multi-echelon supply chain. *International Journal of Logistics Systems and Management, 7*(4), 431-455.

Clark, V. (2002). Sea power 21: Projecting decisive joint capabilities. Retrieved August 3, 2015, from http://www.navy.mil/navydata/cno/proceedings.html

Congressional Budget Office. (2007). *Sea basing and alternatives for deploying and sustaining ground combat forces.* Retrieved August 3, 2015, from https://www.cbo.gov/sites/default/files/07-05-seabasing.pdf

Daim, T. U., & Kocaoglu, D. F. (2008). How do engineering managers evaluate technologies for acquisition? A review of the electronics industry. *Engineering Management Journal, 20*(3), 44-52.

Department of the Army. (2008). *Army regulation 710-3: Inventory management asset and transaction reporting system.* Retrieved August 5, 2015, from http://www.apd.army.mil/jw2/xmldemo/r710_3/main.asp

Department of the Navy. (1994). *Marine Corps warehousing manual.* Retrieved February 15, 2016, from http://www.marines.mil/Portals/59/Publications/MCO%20P4450.7E.pdf

Dillon-Merrill, R. L., Parnell, G. S., Buckshaw, D. L., Hensley, W. R., & Caswell, D. J. (2008). Avoiding common pitfalls in decision support frameworks for Department of Defense analyses. *Military Operations Research, 13*(2), 19-31.

Edwards, W., & Barron, F. H. (1994). SMARTS and SMARTER: Improved simple methods for multiattribute utility measurement. *Organizational Behavior and Human Decision Processes, 60,* 306-325.

Ewing, P. L., Tarantino, W., & Parnell, G. S. (2006). Use of decision analysis in the Army base realignment and closure (BRAC) 2005 military value analysis. *Decision Analysis, 3*(1), 33-49.

Feng, T., & Keller, L. R. (2006). A multiple-objective decision analysis for terrorism protection: Potassium iodide distribution in nuclear incidents. *Decision Analysis, 3*(2), 76-93.

Goodwin, P. & Wright, G. (2004). *Decision analysis for management judgment* (3rd ed.). Chichester, West Sussex, England: Wiley.

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.

Gunderson, S. J., Canfield, M. F., Dann, G., & McCambridge, D. J. (2004). *Naval total asset visibility (NTAV) precisions asset location (PAL): System tests on the SS Curtiss.* Retrieved August 3, 2015 from http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA423570

Joint Chiefs of Staff. (2010). *Joint publication 4-09: Distribution operations*. Retrieved February 15, 2016, from http://www.dtic.mil/doctrine/new_pubs/jp4_09.pdf

Kang, K., & Gue, K. R. (1997). Sea based logistics: Distribution problems for future global contingencies. *Proceedings of the 29th Conference on Winter Simulation*, 911-916.

Keefer, D. L., Kirkwood, C. W., & Corner, J. L. (2004). Perspective on decision analysis applications, 1990-2001. *Decision Analysis, 1*(1), 4-22.

Keeney, R. L. (1992). *Value-focused thinking: A path to creative decision making*. Cambridge, Massachusetts: Harvard University Press.

Keeney, R. L. (2008). Applying value-focused thinking. *Military Operations Research, 13*(2), 7-17.

Keeney, R. L., & Raiffa, H. (1976). *Decisions with multiple objectives: Preferences and value tradeoffs*. New York, NY: Wiley.

Kerchner, P. M., Deckro, R. F., & Kloeber, J. M. (2001). Valuing psychological operations. *Military Operations Research, 6*(2), 45-65.

Kim, S. J., Seo, J. H., Krishna, J., & Kim, S. J. (2008). Wireless sensor network based asset tracking service. *Portland International Conference on Management of Engineering & Technology (PICMET),* 2643-2647.

Kirchhoff, B. A., Merges, M. J., & Morabito, J. (2001). A value creation model for measuring and managing the R&D portfolio. *Engineering Management Journal, 13*(1), 19-22.

Kirkwood, C. W. (1997). *Strategic decision making: Multiobjective decision analysis with spreadsheets*. Belmont, California: Duxbury Press.

Kök, A. G., & Shang, K. H. (2007). Inspection and replenishment policies for systems with inventory record inaccuracy. *Manufacturing & Service Operations Management, 9*(2), 185-205.

Liu, H., Darabi, H., Banerjee, P., & Liu, J. (2007). Survey of wireless indoor positioning techniques and systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 37*(6), 1067-1080.

Mallon, L. G. (2008). *Strategic mobility 21: Joint sea based logistics contractor report 0019*. Retrieved February 15, 2016, from https://www.researchgate.net/publication/235092641_Strategic_Mobility_21_Joint_Sea_Based_Logistics

Mao, G., & Fidan, B. (2009). *Localization algorithms and strategies for wireless sensor networks: Monitoring and surveillance techniques for target tracking*. Hershey, PA: Information Science Reference.

Moore, C. W., & Hanlon, E. (2003). *Sea basing: Operational independence for a new century*. Retrieved August 3, 2015, from http://www.military.com/Content/MoreContent1?file=NI_Sea_0103

Namineni, P. K., Davey, T., Siebert, G., & Jacobus, C. J. (2010). *U.S. Patent No. 7,852,262*. Washington, DC: U.S. Patent and Trademark Office.

National Research Council Committee on Sea Basing. (2005). *Sea basing: Ensuring joint force access from the sea*. Retrieved August 3, 2015, from http://www.nap.edu/catalog/11370/sea-basing-ensuring-joint-force-access-from-the-sea

Naval Research Advisory Committee. (2004). *Sea basing*. Retrieved August 3, 2015, from http://www.nrac.navy.mil/docs/2005_brief_sea_basing.pdf

Oldland, A. R., Golightly, L. K., May, S. K., Barber, G. R., & Stolpman, N. M. (2015). Electronic inventory systems and barcode technology: Impact on pharmacy technical accuracy and error liability. *Hospital Pharmacy, 50*(1), 34-41.

Parnell, G. (2007). Value-focused thinking using multiple objective decision analysis. In A. G. Loerch, & L. B. Rainey (Eds.), *Methods for conducting military operational analysis: Best practices in use throughout the Department of Defense* (pp. 619-656). Alexandria, Virginia: Military Operations Research Society.

Parnell, G. S., Bresnick, T. A., Tani, S. N., & Johnson, E. R. (2013). *Handbook of decision analysis*. Hoboken, New Jersey: Wiley.

Parnell, G. S., Butler, R. E., Wichmann, S. J., Tedeschi, M., & Merritt, D. (2015). Air Force cyberspace investment analysis. *Decision Analysis, 12*(2), 81-95.

Parnell, G. S., Conley, H. W., Jackson, J. A., Lehmkuhl, L. J., & Andrew, J. M. (1998). Foundations 2025: A value model for evaluating future air and space forces. *Management Science, 44*(10), 1336-1350.

Parnell, G. S., Hughes, D. W., Burk, R. C., Driscoll, P. J., Kucik, P. D., Morales, B. L., & Nunn, L. R. (2013). Invited review - survey of value-focused thinking: Applications, research developments and areas for future research. *Journal of Multi-Criteria Decision Analysis, 20*(1-2), 49-60.

Pazour, J., & Meller, R. D. (2011). An analytical model for a-frame system design. *IIE Transactions, 43*, 739-752.

Reilly, P., Pazour, J., & Schneider, K. (2015). *Propagation of unit location uncertainty in dense storage seabasing environments*. Unpublished manuscript.

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.

Scala, N. M., Kutzner, R., Buede, D., Ciminera, C. & Bridges, A. (2012). Multi-objective decision analysis for workforce planning: A case study. *Proceedings of the 2012 Industrial and Systems Engineering Research Conference.* Retrieved February 15, 2016, from http://d-scholarship.pitt.edu/22686/1/Scala_et_al_2012.pdf

Simon, J., Regnier, E., & Whitney, L. (2014). A value-focused approach to energy transformation in the United States Department of Defense. *Decision Analysis, 11*(2), 117-132.

Sullivan, R. (2012). *T-AKE support for small-scale operations ashore: Observations from Coconut Grove 12 - quicklook report.* Quantico, Virginia: United States Marine Corps.

Tong, J., Nachtmann, H., & Pohl, E. A. (2015). Value-focused assessment of cargo value decreasing rates. *Engineering Management Journal, 27*(2), 73-84.

Unal, R., Keating, C. B., Chytka, T. M., & Conway, B. A. (2005). Calibration of expert judgments applied to uncertainty assessment. *Engineering Management Journal, 17*(2), 34-43.

Violino, B. (2005). *The basics of RFID technology*. Retrieved September 24, 2015, from http://www.rfidjournal.com/articles/view?1337

Chapter 4: Proactive vs. Reactive Resource Allocation for Sea-based Logistics

This chapter has been published in the following peer-reviewed journal:

1. Mofidi, Shahab, Pazour, Jennifer A., and Roy, Debit, 2018, "Proactive vs. Reactive Resource Allocation for Sea-based Logistics," *Transportation Research Part E: Logistics and Transportation Review*, 114, 66-84.

Proactive vs. Reactive Order-Fulfillment Resource Allocation for Sea-based Logistics.

Abstract

We study proactive and reactive sea-based order-fulfillment decisions for a set of SKUs. In such systems, a proactive strategy may be more costly than a reactive strategy and variable marginal costs change with respect to an activity profile. We derive the optimal sets of SKUs and their quantities to handle prior (proactive strategy) or after (reactive strategy) demand materializes. Counterintuitive results show the proactive set may not necessarily include the high-demanded SKUs. This work extends the newsvendor model by analyzing negative marginal shortage costs. The model is illustrated with historical data from a sea-based logistics military application.

Keywords

Order-Fulfillment, Newsvendor Model; Negative Marginal Shortage Cost; Proactive and Reactive Strategies, Distribution, Prestaging, Sea-based Logistics

1. Introduction

In military operations, sea-bases consist of a set of ships to "rapidly tailor, deploy, and employ credible, self-sustained forces to respond to a crisis" (US Marine Corps, 2017). This work focuses on the logistics tasks of providing sustainment and resupply materials to troops on the ground from cargo stored in the holds of ships. This scenario, also known as a "Tailored Resupply Packages" scenario, is a proposed approach. While not currently being implemented, it has the advantage to reduce the need to build up logistics assets ashore and can facilitate a more flexible, agile, and responsive supply network for the military operations on the ground (Pazour and Shin, 2016; US Marine Corps, 2017). However, this requires new operational capabilities to transform a set of ships into floating distribution centers capable of conducting order-fulfillment operations from a ship. Similar to the role of an e-commerce fulfillment center, a Tailored Resupply Packages scenario requires fulfilling personalized (tailored) requests for cargo to be delivered quickly to troops on the ground only when requested. That is, emergent requests trigger specific Stock Keeping Units (SKUs) of cargo to be identified, retrieved, and transported from storage area (holds) of ships to the flight deck of these ships. The cargo is then transferred to a receiving vehicle that transports the tailored requests to military operations on the ground. For ease of communication, in this work, we use the term delivery ship to denote the ship that contains the inventory to be requested by the receiving vehicle, which is a connector and can be either other ships or aerial vehicles.

In a sea-based environment, the fulfilment of personalized stochastic requests with little warning are operationally challenging and complex. This is because internal cargo flow processes of the delivery ship require selective offloading of SKUs from very high dense storage environments. Moreover, the ships currently considered as delivery ships (e.g., T-ESD, T-AKR, and T-AKE ships) were not designed specifically for efficient order-fulfillment operations. As displayed in Figure 1, the T-AKE ship is divided into different levels, connected via elevators, and multi-purpose cargo holds that store a variety of cargo. T-AKE ships have the largest cargo-carrying capacity and the largest flight decks of any combat logistics ship afloat and are designed with large open flight decks, which can support aerial delivery and have large areas that can be used to prestage cargo (US Marine Corps, 2017). Material handling processes on the ships are human-intensive, and require manually storing, retrieving, and relocating pallets and containers using forklifts or pallet jacks. Very high density storage requirements in the holds create tight spaces with minimal maneuvering areas and the requirement that some SKUs may need to be moved out of the way to gain access to more deeply stored SKUs. As a result, retrieval efforts in the holds of the ships have degraded performance and are not identical for all SKUs; instead, the retrieval efforts vary for each SKU based on its popularity.

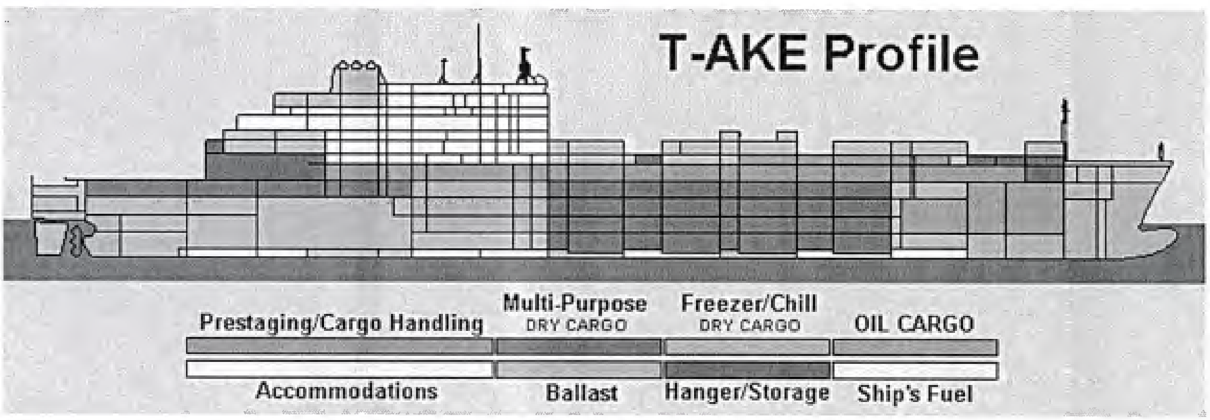


Figure 1. A side-view of T-AKE Dry Cargo Ship; retrieved from http://navy.memorieshop.com/T-AKE/

SKU popularity measure is a commonly used criteria in activity profiling (also known as an ABC analysis) of warehouse operations (Bartholdi and Hackman, 2011). In storage system design, SKUs may be stored based on their popularity measure to minimize overall retrieval efforts. A well-known result from warehouse design is that high-demanded SKUs typically have lower per unit fulfillment costs (Gu et al., 2007). This is because the most popular SKUs are stored in the most convenient locations, while the least popular SKUs are stored in the least accessible and least convenient locations (Hackman et al., 1990). In very high density systems, this effect is even more pronounced because SKUs need to be moved out of the way to gain access to other SKUs (Awwad and Pazour, 2018; Gue, 2006). This results in low-demanded SKUs, which are placed in less convenient locations, requiring much more effort than high-demanded ones.

A responsive sea-based logistics system is one where the receiving vehicle experiences minimal waiting time to receive requested cargo. One way to improve responsiveness to the receiving vehicle, as well as utilization of the transfer process, is to utilize a proactive strategy, which applies resources in anticipation of unknown downstream requests. This is in contrast to a reactive strategy, which applies resources after the request materializes. The concept of "prestaging" cargo on the delivery ship's flight deck before the arrival of the receiving vehicle is a proactive strategy in sea-based logistics. Prestaging involves identifying and retrieving cargo stored in the holds, transporting the cargo to the flight deck of the delivery ship, and temporarily securing and storing cargo on the flight deck in anticipation of requested demand from the receiving vehicle. Whereas, direct transfer is a reactive strategy that involves identifying, retrieving, and transporting cargo from the hold directly to the receiving vehicle, and requires the receiving vehicle to be present at the delivery ship. While prestaging improves the responsiveness to the receiving vehicle, it can increase the labor and storage efforts of the delivery ship. The additional labor efforts are due to additional transportation effort to the designated prestaging locations and double handling of cargo, which is time-consuming due to the need to stabilize cargo on the flight deck, and holding cargo on the flight deck exposes the prestaged cargo to higher risks of damage due to various weather conditions. Also, prestaging may require an additional strike-down process for the excess prestaged cargo.

In sea-based logistics, a requested SKU can be fulfilled through prestaging or direct transfer processes or a combination of both. The tradeoff between these two processes depends on the aggregation of (1) the unit retrieval cost of the SKU in the very high density holds, and (2) the unit double handling and additional transportation costs of prestaging the SKU. For some SKUs, direct transfer is less expensive than prestaging because the effort spent identifying and retrieving the units from holds is less than the additional material handling and stabilization efforts. Other SKUs, especially those stored in the back of holds, require time-consuming retrieval processes, identification and retrieval of units from the holds. Thus, for these SKUs, direct transfer is more costly than the double handling and additional effort of prestaging. Therefore the unit fulfillment costs varies from one SKU to another. We refer to this phenomenon as "varying unit fulfillment costs." In this work, we define a unique cost function that distributes the varying unit fulfillment costs among a set of SKUs proportionally through a SKU's popularity measure.

Given how best to "resupply combat troops from the sea" is an open challenge (Combined Joint Operations from the Sea Center of Excellence, 2012; Parsons, 2013), the goal of this research is to develop models to better understand how to implement prestaging operations for a Tailored Resupply Package scenario, as well as to better understand how the unique features of a sea-based logistics environment impacts fulfilment decisions. This work determines the optimal prestaging policy that minimizes the expected costs of internal cargo flow process for a set of SKUs with skewed, stochastic demand, and varying unit fulfillment costs. Our proposed model determines which SKUs and in what quantities to prestage using a two-stage newsvendor framework, which minimizes additional handling costs of internal cargo flow processes. If the demand that materializes is greater than the prestaging quantity, a reactive strategy is applied to the residual quantity. Based on discussions with the Navy and Marine Corps., a combination of planned and unplanned SKU requests is common in sea-based logistics applications. Planned requests are for SKUs like food, water, or ammo, which are either pushed to receiving ships based on ration calculations, or are requests made in advance by the receiving vehicle (typically in the form of an order). Unplanned requests, which are common, occur because what is requested in advance can change, and additional SKUs are requested when the receiving vehicle arrives. Unplanned requests have high demand uncertainty due to dynamic environments, variability in needs, changing missions, and unplanned maintenance needs. The scope of our work is on the proactive/reactive resource allocation for unplanned SKUs, because a receiving vehicle cannot leave until all requests have been transferred. Thus, the order-fulfillment efforts of unplanned SKUs is a good surrogate for reducing a receiving vehicle's departure time.

This research makes a number of contributions. This is the first work to explicitly model internal cargo flow processes in a sea-based logistics environment and the first to study proactive and reactive sea-based logistics strategies. In doing so, our work captures two unique features of sea-based logistics. First, compared to traditional logistics, sea-based logistics have limited and highly constrained storage space. In very high storage density operations, unit retrieval costs in the holds vary based on a SKU's popularity measure, and thus unit fulfilment costs are different for different SKUs. We explicitly model varying unit fulfilment costs through an activity profile based cost function. The effort to retrieve a SKU from the hold of a ship is allocated through this function. Second, proactive order-fulfillment strategies can sometimes be more costly than reactive strategies. This is because proactive order-fulfillment strategies can require time-consuming double handling and cargo stabilization efforts, additional material handling costs, and additional holding costs (e.g., extra space costs and possible risk of damage when stored on the flight deck). Capturing these two unique features in our proactive resource allocation model resulted in this work also making methodology contributions. Specifically, we extend the two-stage multi-item newsvendor model to consider variable marginal costs for a set of SKUs. Our work is the first to analyze the impact of activity profile based cost functions in a newsvendor modeling framework. In our model, the marginal costs are not constant for different SKUs. That means the tradeoff between conducting a proactive versus a reactive policy changes with respect to the popularity measure of the SKU, which can result in negative marginal shortage costs. Thus, we contribute to the body of newsvendor models by studying the cases where negative marginal shortage costs can happen. Consequently, we derive structural results that include a closed-form expression for the critical point to determine the subset of SKUs to consider as candidates for prestaging (proactive policy). The set of candidate SKUs depends on the application's environment and the relationship between the SKU dependent and independent costs, which for sea-basing is a tradeoff between in-the holds retrieval costs and double handling costs of prestaging. Through a mathematical analysis of structural results, we demonstrate how the set of candidate SKUs varies based on input cost parameters and SKU's popularity measures. We uncover a counterintuitive result; in some situations, prestaging lower-demanded SKUs can outperform a strategy that prestages higher-demanded SKUs. Hence, high-demanded SKUs are not always good candidates for a proactive logistics strategy.

The rest of this paper is organized as follows. Next we review literature on decision making in sea-based logistics, as well as the newsvendor problem and models for reactive versus proactive strategies. We develop the prestaging model and derive structural properties in Section 3. In Section 4, we identify the candidate SKUs amenable for a proactive strategy. In Section 5, we consider how to estimate cost parameter values using a sea-based logistics application. Our models are applied to a case study using data from a naval application in Section 6. We provide conclusions and future research in Section 7.

2. Literature review

This work contributes to two fields of study: decision making in sea-based logistics, and the newsvendor framework. We review these fields in the following subsections.

2.1. Sea-Based Logistics Research

Sea-based logistics has been addressed in the military literature (Beddoes, 1997; Clark, 2002; Combined Joint Operations from the Sea Center of Excellence, 2012; Levin and Friedman, 1982; Parsons, 2013; Willey, 1997). The military research is qualitative in nature, focusing on strategy, policy and requirements planning. There is a limited amount of academic research on mathematical prescriptive and descriptive models to aid decision making in sea-based logistics. Most focus on macro-level supply chain and distribution problems with the objectives of minimizing costs while providing sufficient service levels of support (Pilnick et al., 1991). A simulation based model on the sea-based logistics concept is introduced by Kang and Gue (1997). They simulate supply offloading and determine the allocation of material handling resources for transfer operations. Gue (2003) develops an optimization model for a sea-based distribution system that determines the locations of the sea-based supply units with the minimum need of land-based inventory. He highlights the critical roles of four operational levers on the distribution system design, including the location of the sea base, the inventory held by combat units, available transportation assets, and the timing of troop movements. The developed model can be applied to determine feasibility of a given logistics plan, as well as the development of a real time operational logistics plan. Brown and Carlyle (2008) consider the use of sea-basing for replenishment operations and address capacity planning needs. They formulate an integer linear programing model that maximizes the total utilization of ships and the volume delivered by a specified combat logistics force (CLF). The model, in combination with other organizational tools, can be implemented by fleet planners to conduct comparative analyses. Salmerón et al. (2011) build an optimization mission planner to optimize routes and schedules by maximizing the total mission value earned. This tool provides the user with different plans (e.g., different types of ships), with scenario-specific constraints, and their feasibilities. This tool is more effective than manual planning in regards to the achieved mission value and costs. Also, Qiu and Sharkey (2013) determine both the location and inventory plan of a sea-based logistics operation through integrated dynamic programing algorithms with different service levels and cost factors.

The scope of the sea-based logistics design of all mentioned works is about supply chain network design, transfer processes, inventory, or routing. Cargo is assumed available on the flight deck, neglecting the internal cargo flow process and its influence on response time. One exception is

Futcher (2003), who studies the configuration and density of the cargo storage area and their impacts on the responsiveness for selective offloading in sea-based logistics. He highlights the trade-off between density and retrieval time. In a recent work, Reilly et al. (2017) develop an uncertainty propagation model to quantify the impact of selective offloading policies on search and retrieval processes in dense storage environments for sea-based logistics. Mathematically, they show that high-demanded SKUs have less expected retrieval efforts. For selective offloading in sea-based logistics, the effort and time to find, retrieve, and transport cargo on a ship is significant, especially given the need for very high density storage that requires moving SKUs out of the way to retrieve the needed SKU (Awwad and Pazour, 2018; Gue, 2006; Scala and Pazour, 2016). All of these works only consider retrieval time in the holds and do not consider the full internal cargo flow process to transport cargo to the flight deck.

In summary, the internal cargo flow process is an unexplored aspect of sea-based logistics research. Our work focuses on a Tailored Resupply Package scenario, in which demand is emergent and originating from troops on shore. This is different than the focus of previous work, which was on skin-to-skin replenishment operations, in which demand could be planned in advance (Pazour and Shin, 2016). Consequently, this research contributes to sea-based logistics by studying a sea-based logistics resource allocation problem that captures the impact of internal cargo flow processes in military supply chain decisions. Moreover, this is the first work to study proactive and reactive sea-based logistics strategies. Two unique features of sea-based logistics (in comparison to traditional logistics) are highly constrained storage space, and double handling costs of prestaging. We capture that proactive costs may be more expensive than reactive costs and capture the impact of very high density operations by explicitly modeling a set of SKUs that have varying unit fulfillment cost when being retrieved from the hold of a ship.

2.2. Newsvendor Framework

The newsvendor problem is a classical, well-studied problem in inventory management that determines the optimal order quantity to either maximize the expected profit or minimize the expected cost under stochastic demand. Under traditional assumptions of the newsvendor model, order quantity decisions are made based on a single ordering period. Stochastic demand materializes after the order is placed, which is known as the Single-Period Problem (SPP). For seminal papers and recent reviews of the newsvendor model, see (Arrow et al., 1951; Hillier and Price, 2012; Khouja, 1999; Qin et al., 2011; Silver et al., 1998). If a newsvendor carries more than one type of SKU, the optimal order quantities for different SKUs can be mapped to the multi products newsvendor problem (MPNP). For a comprehensive review of recent MPNP contributions, see Turken et al. (2012).

In the classical newsvendor model, each newspaper costs the newsvendor a unit price c, which then can be sold at a unit price p. At the end of the period, the unsold newspapers will be returned, recycled, disposed or salvaged at a price v, where $v < c < p$. In classical models, the newsvendor has "only one" opportunity to order, which is before demand is realized. An extended version of the newsvendor model is the two-stage newsvendor model, in which the newsvendor has the option of purchasing more newspapers within the selling period if the demand exceeds the initial order quantity. For example, Kodama (1995) provides a model in which a certain level of additional order quantity is allowed during the selling season. Eeckhoudt et al. (1995) examine the behavior of the newsvendor model and compare the impact of cost and price changes on the optimal solution with risk aversion. They also consider that additional newspapers can be purchased and sold at a unit cost $\hat{c}$ where, $v < c < \hat{c} < p$. They conclude that the newsvendor should order more initially if $\hat{c} - c$ is high. However, in their model a change in each of these parameters is considered one at a time, not simultaneously. In our model, we consider the change in both $\hat{c}$ and c simultaneously and for a set of SKUs. This allows us to capture and explore a depreciation in the cost of purchasing additional newspapers (i.e., for some SKUs $c > \hat{c}$). Other names such as "emergency supply option", "limited opportunities to buy" and "resource available options" are also used in the literature to denote an "additional purchase opportunity" for the newsvendor and all assume that $v < c < \hat{c} < p$ (Eynan and Rosenblatt, 1995; Gallego and Moon, 1993; Khouja, 1996; Murray Jr and Silver, 1966; Prasad et al., 2011). While Wang and Webster (2009) mention the cost of shortage to be negative, in their work this represents negative shortage penalty costs and so they do not consider marginal shortage costs that can be negative.

Deciding whether to use a proactive or reactive strategy translates to a two-stage process – with stage 1 being before stochastic demand materializes and stage 2 after demand materializes. Literature exists to address the two-stage newsvendor model (i.e., reactive and proactive operations when additional purchases are allowed) in different environments with diverse names and applications. For example, a similar two-stage problem in inventory management is assemble to order (ATO) and assemble in advance (AIA), in which AIA and ATO are proactive and reactive strategies, respectively. In this type of problem, a manufacturing firm has stochastic demand and is able to carry both raw materials and finished goods. The firm can allocate the finished product's inventory level before the demand materializes, assembling and storing unfinished products in advance. If the demand exceeds the stored inventory, then the firm can employ its assembly line to turn raw materials to finished goods and fulfill the unmet demand. Trade-offs of this allocation problem are the response time versus inventory costs, working in batches versus expedited assembly costs, and chances of lost sales versus salvage costs of the unsold finished products. A single period AIA/ATO problem with stochastic demand for a single SKU is studied by Eynan and Rosenblatt (1995). They derive a profit maximization model with stochastic demand with a critical value similar to the newsvendor model. Then, in order to determine the optimal AIA quantity, a price break is introduced as a threshold. They conclude a single SKU is warranted either full AIA (only proactive) or some AIA and some ATO. Similarly, Chung and Flynn (2001) consider a single SKU production in two stages denoted as anticipatory and reactive. Higher service levels are achieved when a reactive strategy, in addition to a proactive strategy, is employed. Chung et al. (2008) extend the problem by considering a multiple SKU production case, in which the second stage production capacity (reactive capacities) are pre-allocated to each SKU in the first stage (preseason stage) and cannot be changed. They introduce a preseason production threshold and a reactive capacity threshold. Depending on the relative value of the shadow price of the total capacity, they determine if a SKU should be produced only in preseason, only reactively or using a combination of strategies. Reactive production is used as a buffer; hence, the cost of reactive production is always more expensive than the cost of proactive production. In another work, Cattani et al. (2008) study the two-stage newsvendor problem in the fashion industry. They consider a fashion retailer with two products and determine whether to employ a proactive or a reactive production strategy (which they denote as speculative and reactive fabrication) or a combination of both. They conclude that popular SKUs should be produced proactively and less popular SKUs reactively. Similarly, Zhang and Du (2010) consider a multi-item newsvendor problem for a production plant where the production can also be outsourced. Outsourcing given a zero lead time allows the producer to fulfill the unmet demand in the same way as the two-stage newsvendor problem. They assume the second stage costs (outsourcing costs) are more than the first stage costs (in-house production). Reimann (2011) also considers the same assumption of higher second stage production costs in studying the optimal sourcing decisions of a multi-product newsvendor in a speculative production strategy versus anticipatively reserve production capacity strategy with capacity constraints. Products can be categorized as three subsets of strictly speculative production, dual sourcing, and exclusively anticipative capacity reservation. They highlight that the influence of an activity profile is important, but has yet to be studied. Our work is the first to analyze the impact of activity profiling on the optimal solution in a newsvendor modeling framework.

A related inventory management study is Bailey and Rabinovich (2005). While not a newsvendor problem, they do consider a set of SKUs with an activity profile based cost function. They assume a SKU popularity measure to forecast demand for internet book retail orders. They study the tradeoffs between two order fulfilment decisions: (1) in-stock inventory (proactive) strategy, and (2) a drop-shipping (reactive) strategy. Their work suggests that both strategies may be recommended for high-demanded SKUs, and the decision depends on the relationship between cost factors (ordering, reordering, holding, and drop-shipping costs) and the market share. When the market share is high and ordering-costs are low, reactive strategies for high-demanded SKUs are found more beneficial for internet retailers. They study a single-stage inventory model with a constant marginal costs for all SKUs. In our model, marginal costs are variable, respective of the rank of the SKU in the popularity measure, which in turn, result in negative marginal shortage costs. Thus, we contribute to the body of newsvendor models by studying the cases where negative marginal shortage costs can occur.

In contrast to existing work, we formulate a general case of the two-stage newsvendor model with variable marginal costs for a set of SKUs with respect to its rank in the activity profile. Proactive and reactive strategy costs are dependent on the mix of SKUs and their activity profiles. We are motivated by order-fulfillment tasks in facility logistics – which studies movement within a facility – and study uncapacitated problems where flexibility for capacity expansion exists (little or no investments needed), and the main concern is which SKUs to apply resources to in advance of demand and for which to wait until demand materializes. However, facility logistics varies from manufacturing and transportation problems because proactive strategies can require additional effort (in the form of doubling handling, additional material handling, and space), and retrieval efforts can vary by a SKU's popularity measure, specifically, in dense storage environments. In all existing models, whether a single or multi product problem, it is assumed that the proactive strategy is always less expensive than a reactive strategy. In our model, the costs required to fulfill a request using a proactive strategy can exceed the costs required to fulfill the request using a reactive strategy (i.e., $c > \hat{c}$ is possible). This requires a new methodology to handle negative marginal shortage cost in a newsvendor framework. We study the structure of the critical value function and determine its influence on optimal fulfillment strategies.

3. Prestaging Model

We model the expected costs of conducting either prestaging or direct transfer with the flows shown in Figure 2. Prestaging one unit more than the receiving ship's demand will increase the delivery ship's costs due to additional travel, double handling efforts, and cargo stabilization. Moreover, when a SKU is prestaged, but not demanded, it is returned back to the holds because of the variability in the sea state (as items held on the flight deck are exposed to weather conditions and can fall off). On the other hand, prestaging one unit less than the demand may require additional effort or resources to execute expedited processes needed to reduce the idling time of the receiving vehicle. Thus, the order-fulfillment environment experienced by the delivery ship has overage and underage costs, making the two-stage multi-product newsvendor model applicable.

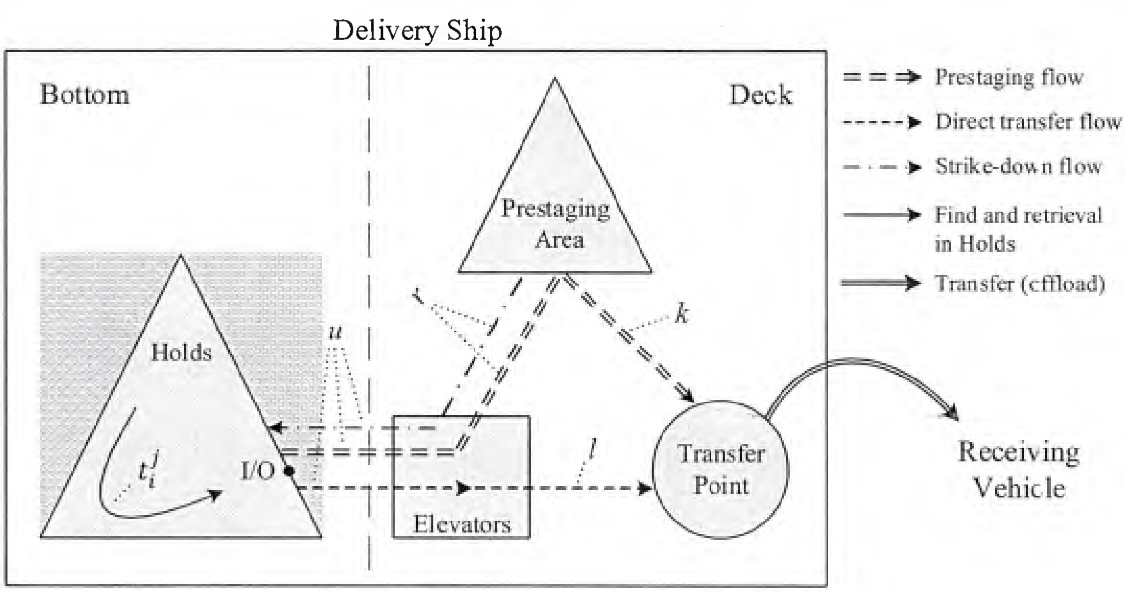


Figure 2, Travel paths between different stations on the delivery ship and their associated costs

The decision maker faces stochastic demand for a set of SKUs stored in the holds indexed by $i \in I$ in the finite set $I = \{1,2,3,\dots,N\}$. The quantity demanded of SKU i is a random variable, denoted as x_i. The quantity demanded of each SKU i is independent of other SKUs' demand. Let $f_i(.)$ and $F_i(.)$ characterize the probability density function and cumulative distribution for x_i, respectively. Refer to Table 3 for notations used in the model.

In the first stage, the decision maker has to decide which SKUs to prestage and in what quantity. Thus, the decision variables of interest are Q_i, which denotes the prestaging quantity for SKU i. If $Q_i > 0$, SKU i is prestaged before the receiving vehicle arrives, otherwise, it is not. Let C_i^1 be the cost to prestage one unit of SKU i in the first stage in anticipation of uncertain demand. In the second stage, when the receiving vehicle arrives, the stochastic demand x_i for SKU i is realized. If the demand x_i exceeds Q_i, the unmet demand $x_i - Q_i$ is directly transferred from the holds at the cost of C_i^2 per unit of SKU i. We assume that all demand is met either via the first or the second stage. Resources needed to fulfill demand, whether via a proactive or reactive strategy, are assumed uncapacitated. In sea-based applications, proactive and reactive capacity expansion is flexible, because military personnel are stationed on the ship. Also, the cargo is prestaged on the flight deck with a large surface area. Hence, the number of SKUs that can be prestaged is typically unconstrained. Moreover, because the receiving vehicle can depart only after all of its requests are fulfilled, no priority exists for the sequence of order-fulfillment of different SKUs. Thus, we assume the benefit of fulfilling different SKUs are independent and constant for all SKUs. If the prestaged quantity received initially, Q_i, exceeds the demand for SKU i, x_i, all unsolicited prestaged SKUs $(Q_i - x_i)$ are assumed returned (salvaged) for a constant value of v per unit, where $v < C_i^2$ and $v < C_i^1$.

To derive cost functions in both stages (i.e., C_i^1 and C_i^2), we capture the internal cargo flow processes (see Figure 2) and then generalize these cost functions. The internal cargo flow processes of the delivery ship consist of two material handling cost components, measured in cost units: (1) SKU dependent costs and (2) SKU independent costs. SKU dependent costs are the ones that vary for different SKUs. In the sea-based logistics application, SKU dependent costs occur in the hold. In very high density storage, the costs spent identifying and retrieving a SKU from the hold and transferring it to the input/output (I/O) point depends on the convenience of the storage location and the quantities of the SKU stored, which depend on the SKU popularity measure. Thus, unit retrieval costs (t_i^1 in stage 1, and t_i^2 in stage 2) are SKU dependent costs, which are estimated based on an activity profile. On the other hand, SKU independent costs are constant for all SKUs regardless of their popularity measure. SKU independent costs in the sea-based application are as follows: strike-up cost u, which includes the unit travel cost from the I/O point of the hold to the elevator, and to the flight deck. r is defined as the unit travel cost on the flight deck from the elevator to the prestaged area, which includes the cost to drop-off, stabilize and secure the load. k is the unit travel cost on the flight deck from the prestaged area to the transfer point. l is the unit travel cost on the flight deck from the elevator to the transfer point. The unit strike-down cost (cost to transfer an unused prestaged SKU from the prestaging area back to the hold) is assumed equal to the summation of u and r. The unit strike-down cost is also derived as $v = -(u + r)$, assumed independent of the type of the SKU. The negativity of the salvage value is because strike-down processes consume resources, which results in any unused units prestaged being a cost and not a benefit. Considering the internal cargo flow process of the delivery ship (as shown in Figure 2), we derive the unit prestaging cost $C_i^1 = t_i^1 + u + r + k$ and unit direct transfer cost $C_i^2 = t_i^2 + u + l$ of SKU i. To summarize, we map the parameters of the order-fulfillment environment in the sea-based application to the parameters in the traditional Newsvendor problem in Table 1.

Table 1. Mapping between the traditional newsvendor model and the sea-based application

Traditional Newsvendor Parameters	**Sea-Based Problem Parameters**
Unit purchase cost in the first stage	Unit prestaging cost
Unit purchase cost in the second stage	Unit direct transfer cost
Unit salvage value	Unit Strike-down cost
Unit selling price	Benefit of fulfilling a request (constant regardless of fulfilled in stage 1 or stage 2).
Traditional Newsvendor Decision Variables	**Sea-Based Problem Decision Variables**
Purchase (order) quantity in first stage	Prestaging quantity

We capture the impact of very high density operations by explicitly modeling a set of SKUs that have varying efforts when being retrieved from the hold of a ship. This work considers a cost function for multiple SKUs based on activity profiling to distribute the unit retrieval costs to different SKUs proportionally to their demand. Activity profiling is the analysis of ranking (profiling) SKUs based on different criteria. The popularity measure is a common way of activity profiling in warehouse operations (Bartholdi and Hackman, 2011), which ranks all SKUs based on their popularity (e.g., number of times requested in a past interval of time). In this research, we define a SKU popularity measure, $g(i)$, which ranks all N SKUs in a monotonically decreasing order by the fraction of times each SKU is requested. For example, $g(j) = 0.4$ means that SKU j is requested 40 percent of the times a receiving vehicle arrived. The monotonically decreasing order means $i = 1$ is the SKU with the highest popularity and SKU $i = N$ has the lowest popularity (i.e., $g(i) > g(i+1)$). We assume $g(i)$ follows the power-law distribution. This assumption is widely used in practice as it models the phenomena where a few SKUs represent a high percent of the activity (e.g., 80% of the requests are from 20% of the SKUs) (Bender, 1981).

A continuous representation of $g(i)$ is given in (1) and is determined as the continuous function in the domain $I^c = \{i \in \mathbb{R} : 0 < i \leq N\}$. Note that the set of SKU I is an integer subset of I^c (i.e., $I \subset I^c$).

$$g(i) = \omega i^{-\gamma}, \qquad \text{where } \omega = \frac{1-\gamma}{N^{1-\gamma} - \gamma} \tag{1}$$

The cumulative fraction $G(i)$ is calculated as:

$$G(i) = \frac{i^{1-\gamma} - \gamma}{N^{1-\gamma} - \gamma} \tag{2}$$

Where, $0 \leq G(i) \leq 1$. This means that $100\ i/N$ percent of the total SKUs accounts for $100G(i)$ percent of the total requests. Details on the derivation of $G(i)$ are provided in Appendix A. Since $\frac{\partial g}{\partial i}(i) < 0$ and $\frac{\partial^2 g}{\partial i^2}(i) > 0$ for $i \in I^c; \gamma \neq 1$ and $\gamma > 0$, the $g(i)$ function is strictly decreasing as a function of i. As the index of SKU i increases, the value of $g(i)$ decreases at a decreasing rate. The exponent γ represents the skewness and can be estimated from the firm's historical data using different methods (e.g. Maximum likelihood, Kolmogorov-Smirnov, Linear regression on log-log PDF). Table 2 displays different values of the exponent value γ for common skewness levels.

Table 2. Values of γ for common skewness

	Skewness			
N	**20/90**	**20/80**	**20/70**	**20/60**
100	1.454	1.174	0.977	0.806
1000	1.201	1.006	0.860	0.727
10000	1.098	0.941	0.818	0.701

Figure 3 illustrates a 20/80 curve for the popularity measure of a set of SKUs $i \in I^c$ and $N = 100$, where both the SKU popularity measure function $g(i)$ and cumulative fraction $G(i)$ versus indices of ranked SKUs are displayed.

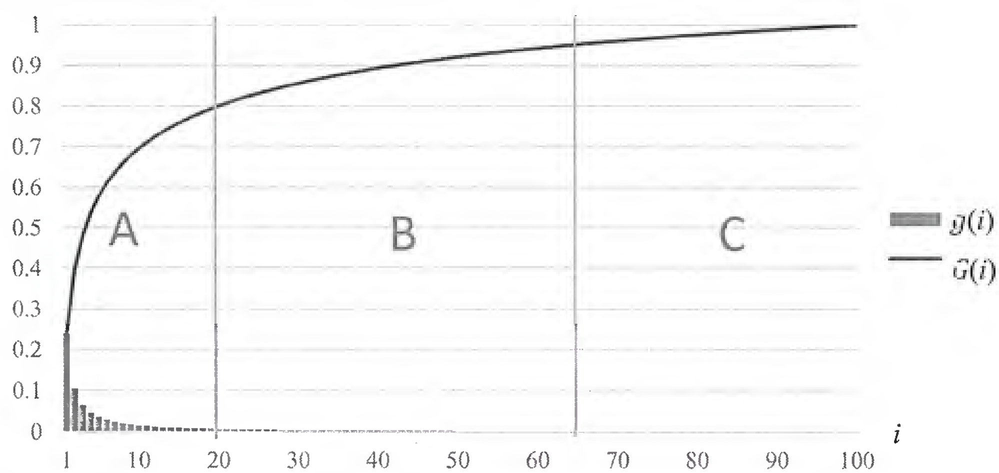


Figure 3. Popularity measure with a 20/80 curve (γ =1.14). The fraction of times SKU i is requested (the SKU popularity measure $g(i)$) and the cumulative fraction $G(i)$ for $i = [1,100]$. High, mid and low-demanded SKUs are represented as A, B and C classes of SKUs, respectively.

In this model, we assume t_i^j, the unit retrieval cost of SKU i in stage $j \in \{1,2\}$, or unit SKU dependent cost in general, is inversely proportional to the SKU's popularity measure (i.e., $g(i)$ as defined earlier). Hence, t_i^j will increase as $g(i)$ decreases. This assumption follows a well-known result from the warehouse design that high-demanded SKUs typically have lower per unit fulfillment costs (Gu et al., 2007). It is consistent with results in the work of Reilly et al. (2017) that show for very high density storage systems, high-demanded SKUs have less expected retrieval efforts. This relationship is illustrated in (3).

$$t_i^j = {\hat{t}^j}/{Ng(i)}, \qquad \begin{array}{l} \forall i \in I, g(i) > 0\,, \\ j \in \{1,2\}, \\ \hat{t}^j \geq 0 \quad and \quad t_i^j \geq 0 \end{array} \tag{3}$$

In Equation (3), $\hat{t}^j$ can be thought of as the average unit retrieval cost from the hold in stage $j \in \{1,2\}$, which is a constant for all SKUs. By replacing $g(i)$ from Equation (1) into (3),

$$t_i^j = \beta_j\, i^\gamma, \qquad \begin{array}{l} \forall i \in I, \quad j \in \{1,2\}, \\ \text{where } \beta_j = {\hat{t}^j}/{N\omega} \end{array} \tag{4}$$

To develop a tractable model, a popularity measure (ABC curve) is fit to the historical data of the SKUs stored in the holds and a specific value of γ is extracted by (2) that captures the skewness. In (4), the summation of all costs that are dependent on a SKU's popularity are distributed proportionally to each SKU through i^γ. The multiplier β_j is the average retrieval factor or mean SKU dependent cost factor in stage $j \in \{1,2\}$. In particular, $\beta_j\, i^\gamma$ models the relationship between SKU dependent costs and demand, i.e., SKU dependent costs are lower for high-demanded SKUs and higher for low-demanded SKUs. Two different values can be assigned to the average unit retrieval cost ($\hat{t}^j$) for each proactive and reactive stages due to the different throughput (service rates) or congestion factors that impact costs in each stage. For example, increasing throughput rates may require an increase in retrieval costs in the hold by applying more personnel to speed up tasks. Thus, superscript $j \in \{1,2\}$ represents parameter values for the proactive and reactive stages, respectively.

$$t_i^1 = \beta_1 i^\gamma, \qquad t_i^2 = \beta_2 i^\gamma \tag{5}$$

SKU independent costs incurred by the internal cargo flow process can be generalized as $\alpha_1 = u + r + k$ and $\alpha_2 = u + l$. Thus, α_1 and α_2 represent unit independent costs in stage 1 and stage 2, respectively. Decision makers can extract parameter values depending on the type of the business and the application's environment and classify costs as either SKU independent or SKU dependent costs. Thus, a generalized structure of the unit costs for SKU i with a proactive and reactive strategy are given in (6).

$$C_i^1 = \alpha_1 + \beta_1 i^\gamma, \qquad C_i^2 = \alpha_2 + \beta_2 i^\gamma \tag{6}$$

Due to the SKU popularity measure in Expression (1), the unit cost is an increasing function of SKU index i. However, the unit cost functions are influenced by the given values of α_1, α_2, β_1 and β_2, which are all assumed to take on values greater than 0. Both SKU independent and SKU dependent costs can vary whether the request for a SKU is fulfilled using a proactive versus a reactive strategy. For example, if reactive processes require expedited services for SKU dependent costs, then $\beta_2 > \beta_1$. However, for SKU independent costs, if a proactive process requires additional resources (such as extra space or extra resources due to double handling), then $\alpha_1 > \alpha_2$. A unit marginal shortage cost is the difference between the direct transfer and prestaging costs (i.e., $C_i^2 - C_i^1$). Since $C_i^1 = t_i^1 + u + r + k$ and $C_i^2 = t_i^2 + u + l$, the unit marginal shortage cost can be calculated as $C_i^2 - C_i^1 = t_i^2 - t_i^1 + l - (r + k)$. Due to different potential values of parameters, the unit cost in stage one to stage two for particular SKUs might increase while it decreases for other SKUs (i.e. for some SKUs $C_i^1 > C_i^2$ and for other SKUs $C_i^2 > C_i^1$), as shown in Figure 4. This results in variable marginal shortage costs, which means that the unit marginal shortage cost varies for each SKU i. Similarly, the unit marginal overage cost is calculated as $C_i^1 - v = t_i^1 + 2(u + r)$, which also varies for different SKUs. A change in the environment, may affect the order-fulfilment decision to recommend a different SKU for prestaging. For example, a change in hold density (i.e., lessen the density and thus lessen the retrieval and search effort) will lead to a decrease in retrieval cost. Thus, the marginal shortage cost reduces overall that will, in turn, favor more high-demanded SKUs (popular ones) to be excluded from candidates for prestaging policy. More details on the changes in cost functions and their consequences on the order-fulfillment decisions are provided in Section 4.1 followed by Section 5, which provides specifics for sea-based logistics applications.

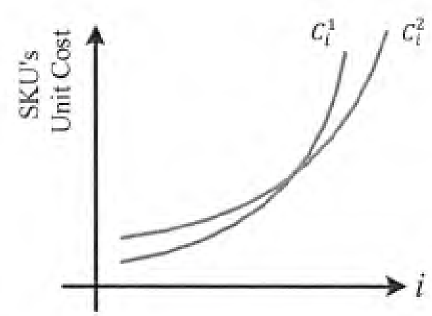


Figure 4. Different SKU's unit costs in proactive and reactive strategies, C_i^1 and C_i^2 respectively.

3.1. Optimal prestaging quantity

Table 3 summarizes the variables and parameters used in this paper.

Table 3. List of parameters and variables

N	the number of SKUs
i	the index of SKU $i \in I = \{1,2,\dots,N\}$
x_i	the quantity demanded of SKU i, a random variable
f_i	the probability density function for x_i
F_i	the cumulative distribution for x_i

$g(i)$	the SKU popularity measure function, which denotes the fraction of times SKU i is requested by a receiving vehicle.
$G(i)$	the cumulative fraction of times that SKUs rank i or lower are requested.
α^j	the SKU independent costs incurred by the internal cargo flow process in stage $j \in \{1,2\}$
β_j	the average retrieval factor (mean SKU dependent cost factor) in stage $j \in \{1,2\}$
t_i^j	the unit retrieval cost of SKU i in stage $j \in \{1,2\}$
$\hat{t}^j$	the average unit search and retrieval cost from the hold in the stage $j \in \{1,2\}$
C_i^j	the unit cost of transferring SKU i from a hold to the transfer point in stage $j \in \{1,2\}$
Q_i^*	the optimal prestaging quantity of SKU i
u	the cost of transferring one unit of cargo from a hold to an elevator
r	the cost of transferring one unit of cargo from an elevator to the prestaging area
k	the cost of transferring one unit of cargo from the prestaging area to the transfer point
l	the cost of transferring one unit of cargo from an elevator to the transfer point
v	the strike-down cost of one unit of cargo

Considering the varying unit fulfillment costs of a proactive and reactive strategy, the decision maker's total cost function for the total prestage quantity Q when the total demand of x is realized is as follows:

$$Z(Q,x) = \sum_{i=1}^{N} Z_i(Q_i, x_i) \tag{7}$$

An optimal prestaging policy determines the values for Q_i for all $i = 1,2,\dots,N$ that minimize (7). The demand quantity of SKU i, x_i, is assumed independent of the other SKUs, and flexibility exists such that capacity constraints are not binding. Therefore, this risk-neutral inventory problem can be decomposed into independent optimal policies for each SKU. Hence, the total cost function for SKU i is as follows:

$$Z_i(Q_i, x_i) = \begin{cases} C_i^1 Q_i - v(Q_i - x_i), & Q_i \geq x_i \\ C_i^1 Q_i + C_i^2(x_i - Q_i), & Q_i < x_i \end{cases}$$

Or equivalently,

$$Z_i(Q_i, x_i) = C_i^1 Q_i - v(Q_i - x_i)^+ + C_i^2(x_i - Q_i)^+ \tag{8}$$

The expected value of the cost function is analyzed to obtain the optimal prestage quantity Q_i^* for each SKU, given that the demand quantity of SKU i, x_i, is a stochastic variable.

Hence, the objective is:

$$\min \Pi_i(Q_i) \equiv E[Z_i(Q_i, x_i)]$$

$$\Pi_i(Q_i) = C_i^1 \mu_i + [(C_i^1 - v)E(Q_i - x_i)^+ + (C_i^2 - C_i^1)E(x_i - Q_i)^+] \tag{9}$$

Where $E[.]$ denotes the expectation operator and μ_i the mean demand quantity of SKU i.

The first-order condition for (9), using the Leibnitz rule, is:

$$\frac{\partial \Pi_i(Q_i)}{\partial Q_i} = (C_i^1 - v)\int_0^{Q_i} f_i(x_i)dx_i + (C_i^2 - C_i^1)\int_{Q_i}^{T} -f_i(x_i)dx_i = 0$$

Setting the first order conditions (FOC) equal to zero, results in:

$$(C_i^1 - v)[F_i(Q_i)] - (C_i^2 - C_i^1)[1 - F_i(Q_i)] = 0$$

$$F_i(Q_i) = \frac{C_i^2 - C_i^1}{C_i^2 - v} \quad (10)$$

The expected cost function for SKU i, is convex (see Appendix B). Hence, the minimum value for (9) is calculated by setting the FOCs equal to zero. The optimal prestage quantity for SKU of index i (Q_i^*) that minimizes (9) is determined by Equation (11).

$$Q_i^* = \min\{Q_i \in W | F_i(Q_i) \geq CV(i)\}, \quad W = \{0,1,2,3,\ldots\} \quad (11)$$

$$CV(i) = \frac{C_i^2 - C_i^1}{C_i^2 - v}, \quad \forall i \in I \quad (12)$$

Due to the different potential values for the parameters used in Equation (12), $CV(i)$, which denotes the critical value, is unconstrained in sign. However, from Equation (11), Q_i^* is mapped to SKU i's demand cumulative distribution function, F_i. From probability theory, the cumulative distribution function must be between zero and one (i.e., $0 \leq F_i(.) \leq 1$) and in Section 3.3, we discuss the different values of $CV(i)$ that can occur and their associated $Q_i^{*\prime}s$.

3.2. Properties of $CV(i)$

As the solution of (11) depends on the value of $CV(i)$, we study in this section how $CV(i)$ behaves as a function of SKU i. Specifically, we characterize the properties of $CV(i)$ and provide relevant proofs. In Section 4, these properties will be used to identify bounds on SKUs that are considered for prestaging.

Property 1. The codomain of $CV(i)$ contains negative values

Since $C_i^2 > v$, the denominator of $CV(i)$, as defined in (12), is strictly positive for all SKUs. Therefore, the $CV(i)$ function is continuous and differentiable in the domain I^c. As shown in Figure 4, different input parameters of the cost functions C_i^2 and C_i^1 mean for some SKUs $C_i^2 > C_i^1$ while for others $C_i^2 < C_i^1$. Negative values in the numerator in (12) can occur. As mentioned, this is known as negative marginal shortage costs because (12) can be derived by replacing $C_i^2 - C_i^1$ as the marginal shortage cost and $C_i^1 - v$ as the marginal overage cost in the newsvendor problem's well known critical value. In Appendix C, we prove the numerator of $CV(i)$ will always be less than the denominator; therefore, the codomain of $CV(i)$ is as follows:

$$CV(i) \in (-\infty, 1) \quad \forall i \in I^c \quad (13)$$

To fully understand how $CV(i)$ behaves as a function of i and takes values in this codomain, we provide the critical value definition in (14), which substitutes the definition given in Equation (12) for C_i^2 and C_i^1 . Therefore,

$$CV(i) = \frac{(\beta_2 - \beta_1)i^\gamma + \alpha_2 - \alpha_1}{\beta_2 i^\gamma + \alpha_2 - v} \tag{14}$$

Due to (14), $CV(i)$ is a rational function of i (i.e., the ratio of two polynomials of i). We define the cost functions in the continuous domain I^c and continue to study the properties of $CV(i)$ as a continuous function in this section. We are specifically interested in Property 2 to obtain bounds on $CV(i)$.

Property 2. Asymptotes

As the value of i grows, $CV(i)$ approaches a certain value, s, which is the horizontal asymptote for this function given in (15).

$$s = \lim_{i \to \infty} CV(i) = \frac{\beta_2 - \beta_1}{\beta_2} = 1 - \frac{\beta_1}{\beta_2} \quad \forall i \in I^c \tag{15}$$

The asymptote s is a bound on the function $CV(i)$ in our model. Given the asymptote value only depends on the value of two parameters $\beta_1$and β_2, we use this property in Section 4 to determine the optimal set of SKUs to prestage (see Theorem 2). Also, the critical value $CV(i)$ has no vertical asymptote, therefore, it is continuous in the domain I^c.

Property 3. Monotonic function

$CV(i)$ is a continuous monotone function of i in the domain I^c.

Proof: The first derivative of $CV(i)$ with respect to i is calculated as:

$$\frac{\partial CV(i)}{\partial i} = \frac{\gamma \varphi i^{\gamma - 1}}{(\beta_2 i^\gamma + \alpha_2 - v)^2}, \tag{16}$$

Where,

$$\varphi = (\alpha_1 - v)\beta_2 - (\alpha_2 - v)\beta_1 \tag{17}$$

Since $\gamma > 0$ and $i \in I^c$, the sign of (16) only depends on the value of φ. Specifically,

$$\frac{\partial CV}{\partial i}(i) = \begin{cases} \geq 0 & \text{if } \delta \geq 0 \quad \rightarrow CV(i) \text{ is monotonically increasing in } I^c \\ < 0 & \text{if } \delta < 0 \quad \rightarrow CV(i) \text{ is strictly decreasing in } I^c \end{cases} \tag{18}$$

Therefore, (18) shows the $CV(i)$ function is a continuous monotone function in the domain I^c. This is illustrated in Figure 5.

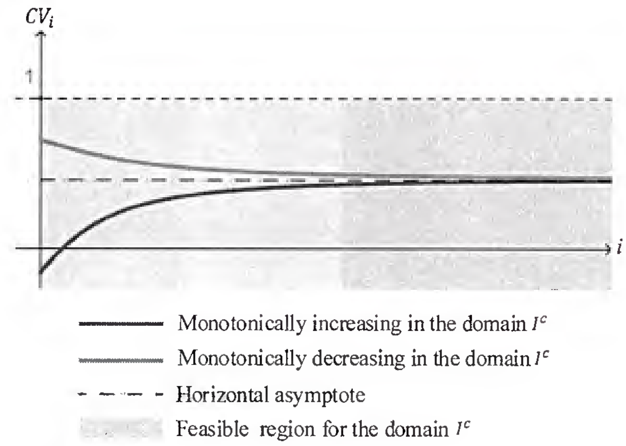


Figure 5, Monotonic $CV(i)$ function in the domain I^c.

From (11), Q_i^*, the optimal prestage quantity for SKU i can only take on positive values when the critical value becomes positive. Using property 3, the $CV(i)$ function is decomposed into two parts where $CV(i) > 0$ and $CV(i) \leq 0$. Therefore, considering these two cases and the bounds on $CV(i)$ defined in (13), the optimal prestage quantity Expression (11) can be decomposed into two parts in (19).

$$Q_i^* = \begin{cases} F_i^{-1}(CV(i)), & \text{if} \quad 0 < CV(i) < 1 \\ 0 & \text{if} \quad CV(i) \leq 0 \end{cases} \tag{19}$$

For the case where the $CV(i)$ function is monotonically decreasing, the value of $CV(i)$ can become negative for some SKUs and due to (19), the optimal prestage quantity (Q_i^*) becomes zero. In Section 4, we investigate these situations that result into structural bounds for the optimal solution.

4. Candidate SKUs for Prestaging in Stage 1

To answer the first research question (which SKUs to apply a prestaging strategy), we identify which SKUs are disadvantageous to prestage. These SKUs are assigned a prestage quantity of zero ($Q_i^* = 0$). The remaining SKUs are the ones, if prestaged in the first stage, have the potential to be beneficial. We call such SKUs as the Candidate SKUs for Prestaging (CSP). Therefore, CSP is a subset of the domain (i.e., CSP $\subseteq I^c$) containing the index of SKUs to be considered for prestaging in the first stage. The SKUs belonging to the set of CSP are considered, but are not guaranteed to have a positive value for Q_i. (i.e., $Q_i^* \geq 0$ for $i \in$ CSP and $Q_i^* = 0$ for $i \notin$ CSP).

4.1. Critical point

As shown in Equation (19), if the critical value of SKU i becomes negative, $Q_i^* = 0$ Due to property 3, $CV(i)$ is a monotone function of i in the domain I^c. Therefore, to define the set of CSP, we are looking for the critical point, which we denote as θ, such that $CV(\theta) = 0$ and $\theta \in I^c$.

Theorem 1. If there exists a $\theta \in I^c$, such that $CV(\theta) = 0$, then θ is unique and can be found by (20).

$$\theta = \left(\frac{\alpha_1 - \alpha_2}{\beta_2 - \beta_1}\right)^{\frac{1}{\gamma}}, \qquad \text{where } \gamma \neq 1 \text{ and } v > 0 \quad (20)$$

Proof is provided in Appendix D. Theorem 1 states that if there exists a critical point θ in the domain I^c, then the critical point will divide the set of I^c into two disjoint sets: one set will contain the SKUs i with positive values of $CV(i)$ and one set will contain the SKUs with negative values of $CV(i)$. The set with positive values of $CV(i)$ becomes the CSP set. If such θ cannot be found, the I^c is not decomposed. Instead CSP either includes all SKUs $i \in I^c$ or does not include any SKU $i \in I^c$. According to (20), θ can be calculated as a radical expression with the skewness exponent γ. Therefore, a straightforward feasible real root solution for θ in the domain I^c can be found only if the radicand is positive. Otherwise, the solution is either not feasible ($i.e., \ \theta \leq 0$) or there does not exist a θ such that $CV(\theta) = 0$. It is important to note that the solution to (20) yields the critical point θ, but is not solely enough to determine the CSP set. Different potential values of cost parameters (α_1, α_2, β_1 and β_2) impact the behavior of the $CV(i)$ function, thus θ value, and ultimately the CSP set.

Theorem 2. The cost parameters, the sign of the asymptote s, and the critical point θ are required to define the CSP set.

Proof is provided in Appendix E. Table 4 summarizes the possible combinations.

Table 4, the determination of CSP interval

Asymptote	Condition	CSP	Figure
s > 0	if θ is feasible	$(\theta, N]$	6a
	o.w.	I^c	6b and 6c
s = 0	if $\alpha_2 > \alpha_1$	I^c	6f
	o.w.	$\emptyset$	6d and 6e
s < 0	if θ is feasible	$(0, \theta)$	6i
	o.w	$\emptyset$	6g and 6h

According to Theorem 2, if s > 0 and if θ is feasible, SKUs with low indices (high-demanded SKUs) do not have the potential to be considered for prestaging and should be directly transferred once the demand materializes. However, SKUs with $i > \theta$ do have the potential to be considered for prestaging. See Figure 6a. Otherwise, if a feasible θ cannot be found, all SKUs have the potential to be considered for prestaging (i.e., $Q_i^* \geq 0$, $\forall i \in I^c$). See Figures 6b and 6c.

If s < 0 and if θ is feasible, SKUs with higher indices (low-demanded SKUs) than θ are not beneficial to be considered for prestaging, but SKUs with lower indices than θ should be considered for prestaging. See Figure 6i. Otherwise, when θ is not feasible, no SKUs are considered for prestaging. All SKUs should be directly transferred in the second stage. See Figures 6g and 6h.

If $s = 0$ and if $\alpha_2 > \alpha_1$, all SKUs have the potential to be considered for prestaging. See Figure 6f. On the other hand, when $\alpha_2 \leq \alpha_1$, CSP is a null set, which means none of the SKUs are recommended for prestaging. See Figures 6d and 6e.

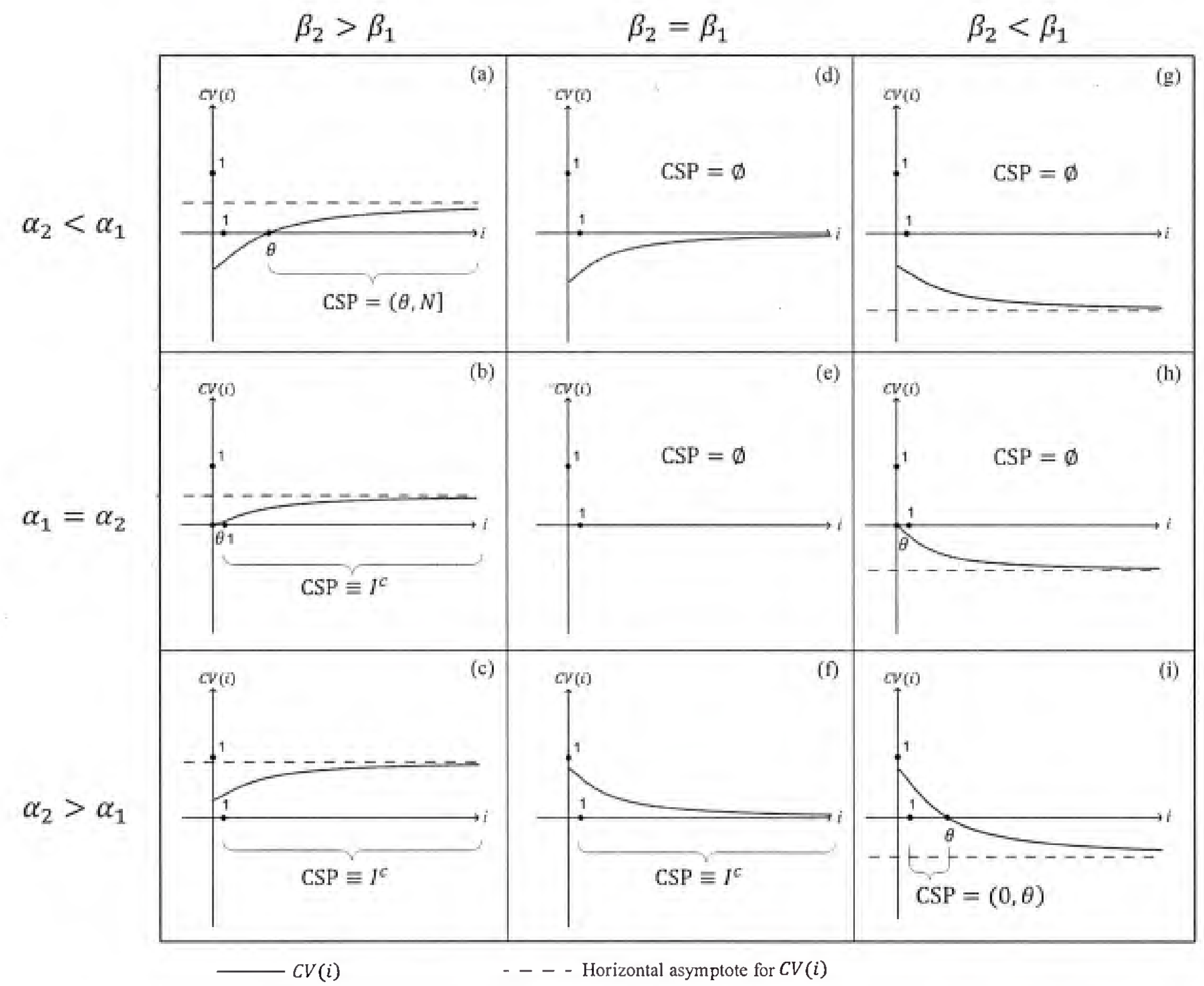


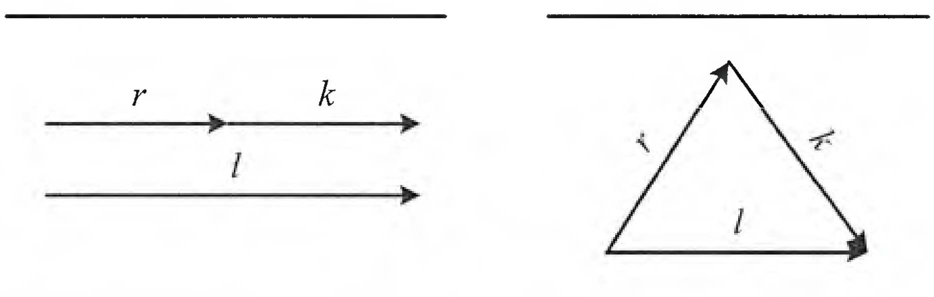
Figure 6. $CV(i)$ function in the domain set I^c for different values of (α_1, α_2, β_1 and β_2). Dotted line is the asymptote (from Property 3).

4.2. Parameters sensitivity analysis

The recommendations of our model are sensitive to the input parameters of the model. If θ exists, the CSP set is a subset of the set of SKUs. The set of SKUs to be considered initially is not always a full set or a null set. For conditions in which θ is feasible in the domain I^c (the ones illustrated in Figures 6a and 6i), the CSP is influenced by θ; thus, we investigate how a change in values of different input parameters influences θ, and the optimal solutions. To do so, we define $\Delta\beta = |\beta_2 - \beta_1|$ and $\Delta\alpha = |\alpha_2 - \alpha_1|$. Due to Expressions (14) and (20), as $\Delta\beta$ approaches zero, θ increases. If this condition occurs in Figure 6a, high-demanded SKUs are excluded from CSP and instead will be directly transferred at stage 2. Whereas in Figure 6i, high-demanded SKUs are included in CSP and will be prestaged at stage 1. As $\Delta\alpha$ approaches zero, θ decreases. When $\Delta\alpha$ approaches zero, more high-demanded SKUs are considered in the CSP interval for the conditions in Figure 6a; contrastingly, fewer high-demanded SKUs will be present in the CSP interval for the conditions in Figure 6i. As N increases, θ decreases but with insignificant change. θ is not sensitive to the value of N. θ is also insensitive to the value of v. This means that the value of the strike-down cost does not affect the CSP interval and thus the SKUs considered in the prestaging do not change. However, the critical value $CV(i)$ is affected by the value of v. A decrease in the strike-down cost results in an increase in the critical value. This is because in the sea-based logistics problem v represents strike-down costs, which are negative salvage values. If the decision maker does not enforce a strike-down process after each receiving vehicle order (e.g., the weather condition is not threatening or the next receiving vehicle is fast approaching), and decides to leave the unused prestaging inventory on the deck in the prestaging area, then v approaches zero. As a result, if a SKU is prestaged, the prestaging quantity would likely be higher in this scenario (because $CV(i)$ is larger, but would depend on the specific distribution of x_i). By conducting the prestaging process for order-fulfillment of more than one receiving vehicle, the decision maker is able to hedge the prestaging risk, and thus likely would prestage more units.

5. Parameter Analysis in Sea-based Application

Whereas the analysis in Section 4 presented general cases based on the relative values of α_1, α_2, β_1 and β_2, in this section, we consider how to estimate these values using a sea-based logistics application. As illustrated in Figure 2, prestaging units in anticipation of a receiving vehicle's demand or waiting for the vehicle to arrive have both SKU independent and SKU dependent costs. In practice, different delivery ships will have different layouts, prestaging locations, and material handling equipment that affect the costs in each stage and the decision of prestaging SKUs. Also, in some transfers, the response time of the delivery ship may be less important given the available time of the receiving vehicle. These different conditions are implemented by changing cost parameters in our model, which result in four different cases summarized in Table 5 and described below.



a. prestaging path without detour ($\alpha_2 - \alpha_1 = 0$) b. prestaging path with detour ($\alpha_2 - \alpha_1 < 0$)

Figure 7, a) prestage area is on the shortest path between the elevator and the transport point (case 1) b) prestaging area is somewhere that requires additional efforts (case 2)

(Case 1) When responsiveness is a concern and prestaging is on the shortest path, all SKUs (from the most popular to the least popular ones) are considered for prestaging.
In this case, the delivery ship is concerned with both anticipatory labor efforts of the delivery ship and responsiveness of the receiving vehicle. Therefore, the decision maker is willing to spend more labor resources to decrease the response time. Thus, the mean SKU dependent cost factor in the hold increases in the second stage (i.e., $\beta_2 - \beta_1 > 0$). Moreover, the prestage area is located on the shortest path between the elevator and the transport point; there is no detour (see Figure 7a). Thus, the process required to prestage SKUs does not require additional effort versus bringing the SKUs directly from the hold (i.e., $r + k = l$). This means the SKU independent costs are identical (i.e., $\alpha_1 = \alpha_2$). This case maps to Figure 6b, where $CV_i > 0$ for all SKUs. Therefore, every SKU is a candidate for prestaging if the double handling effort is assumed negligible. Since the prestaging area is in on the shortest path of the transferring path, the delivery ship can prestage cargo before the receiving vehicle arrives without additional strike-up effort; however, the unrequested SKUs on the deck still have to be returned to the hold. Even though all SKUs are considered for prestaging, many of the low-demanded SKUs will not be prestaged due to their low demand probabilities (i.e., $CV(i) \leq F_i(Q_i = 0) \rightarrow Q_i^* = 0$).

(Case 2) When responsiveness is a concern and the prestaging path creates a detour, high-demanded SKUs are excluded from candidates for prestaging.
In this case, the delivery ship has the same concern as Case 1, but this time the prestaging area is located somewhere that creates a detour, requiring additional costs (i.e., $r + k > l$). This results in $\alpha_1 > \alpha_2$. See Figure 7b. For this case, the effort to fulfill a request from demand for a SKU that has been prestaged requires more effort because of increased travel and double handling, than fulfilling the demand for the SKU directly from the hold. Here both responsiveness and costs are considered such that the dependent cost factor in the second stage is increased to cope with the responsiveness (i.e. $\beta_2 > \beta_1$). However, in this environment, a SKU that is prestaged requires an extra travel costs in comparison with an SKU that is brought directly from the hold to the receiving vehicle reactively, i.e., $\alpha_1 > \alpha_2$. This case maps to Figure 6a, where CV_i remains unconstrained and those SKUs are recommended that have the index $\theta < i \leq N$.

(Case 3) When the responsiveness is ignored, none of the SKUs are prestaged.
In this case, the delivery ship is concerned only with its own efficiency. Thus, prestaging decisions are made based on the lowest possible labor effort, ignoring the responsiveness needs of the receiving vehicle. In other words, the decision maker is not willing to apply more resources to increase the throughput in the second stage to compensate for the delay in retrieval due to the time spent in the holds. Thus, the mean SKU dependent cost factors are identical (i.e., $\beta_2 - \beta_1 = 0$). Also, the prestaging location is inconsequential (i.e., $\alpha_1 \geq \alpha_2$). According to Section 4.1, setting $\beta_2 = \beta_1$ and if $\alpha_1 \geq \alpha_2$, results in CSP $\equiv \emptyset$ since $CV_i \leq 0$ for all SKUs (see Figure 6d and 6e). The recommendation is trivially not to prestage any SKU because in this case prestaging requires more effort for all SKUs.

(Case 4) When the presence of the receiving ship causes congestion in path ways, all SKUs are considered for prestaging.
In this case, congestion causes the direct transfer path to be impaired. This means that the travel costs associated with prestaging are less than the direct travel from holds to the transfer point (i.e., $r + k < l$). This can happen when the travel cost in the second stage (α_2) is negatively affected by blocking and congestion, which commonly increases when the receiving vehicle arrives. If the velocity reduction to go from the elevator to the transfer point is reduced enough, then $\alpha_1 < \alpha_2$. In addition, the delivery ship's concern about the responsiveness is inconsequential (i.e. $\beta_2 \geq \beta_1$), then the $CV_i > 0$ for all SKUs. In this case, all SKUs are considered for prestaging. This case maps to Figures 6c and 6f.

Table 5. Recommendations for CSP with respect to the four cases

Case	$\beta_2 - \beta_1$	$\alpha_2 - \alpha_1$	$CV(i)$	*Recommendation*	*CSP Interval*
1	> 0	0	> 0	Candidate all SKUs for Prestaging	$(0, N]$
2	> 0	< 0	$(-\infty, 1)$	No prestaging for $i < \theta$	$(\theta, N]$
3	0	≤ 0	≤ 0	No prestaging for $\forall i \in I$	$\emptyset$
4	≥ 0	> 0	> 0	Candidate all SKUs for Prestaging	$(0, N]$

6. Sea-Based Military Logistics Numerical Examples

In this section, numerical examples using data adapted from a sea-based application are applied to our developed model. These examples are based on using a T-AKE ship, the USNS Sacagawea, as a floating distribution center. To do so, dry cargo stored in the holds are identified, retrieved, and then transported to the flight deck to fulfill demand requests from troops on shore, transported using receiving vehicles. Through the use of load plans of the USNS Sacagawea and subject-matter experts, we extract SKU inventory levels, estimated demand skewness, and costs to fulfill requests for cargo from receiving vehicles either by prestaging cargo or responding once the receiving vehicle arrives. Due to data availability, we assume unit handling was at the pallet level, thus all SKUs in the numerical example are demanded in pallet quantities. However, our model is general and capable of studying systems with smaller handling units. Estimates for the available material handling effort per pallet were extracted from conversations with logistics officers and based on time study data captured through military exercises. We apply the seabasing data to our model and compare the optimal prestaging quantities from our model with the two-stage model newsvendor problem that considers constant marginal costs (which we denote as the constant newsvendor model, CNV for short).

We consider four different holds and effort data, which map to each of the cases mentioned in Section 5. In this example, we capture costs in terms of labor-minutes to conduct order-fulfillment processes. Table 6 displays parameter values for the four cases, as well as results from our model, the number of SKUs in each hold (N), and the total pallet units of inventory stored in the hold (M). Therefore, summing over all four holds, the data set contained a total of 126 SKUs in four separate holds, for a combined 3268 pallets of inventory. To increase storage utilization, cargo are usually stacked densely next to and on top of each other in the hold. Thus, in the holds, retrieving cargo often requires moving SKUs out of the way to gain access to other SKUs. Thus, the processing effort in the holds is time and labor intensive, and also varies based on the number of copies of SKUs stored. High-demanded SKUs take less effort because they are stored in the most convenient locations and are easy to find, whereas low-demanded SKUs are stored in the least accessible locations. The average unit search and retrieval cost in the hold is estimated as $\hat{t}^1 = 20$ labor-minutes in the proactive stage. This parameter is calculated from the total retrieval time divided by total requests fulfilled in the proactive stage without considering the responsiveness factor. When responsiveness is a concern, a 25% increase in labor cost for in the hold effort is considered in calculations resulting in $\hat{t}^2 = 25$ labor-minutes in the reactive stage. For Case 3, $\hat{t}^2 = \hat{t}^1 = 20$ because responsiveness is not a concern and no additional labor is assigned to the hold tasks when a receiving vehicle arrives.

Considering the different equipment, paths, and distances associated with different holds, the unit travel costs (i.e., $u, l, r,$ and k) are extracted and shown in Table 6 in labor-minutes. In our analysis, holds in different locations contain different SKUs. For a given hold, a popularity measure is determined by Expression (2), which results in estimating γ values in Table 6. In addition, the expected total prestage quantity V in pallet units requested from a given hold by the receiving vehicle is assumed to be known, which is the sum of the order quantities of the different SKUs (i.e., $V = \sum_{i=1}^{N} x_i$). This assumption is made because common receiving vehicles (also known as connectors) typically have a fixed carrying capacity, such as with the MV-22B Osprey, CH-53E Super Stallion, UH-1Y Venom, Landing Craft Air Cushion (LCAC), Landing Craft Utility (LCU), Expeditionary Fast Transport (T-EPF), and the Lighter, Amphibious Resupply, Cargo-V (LARC-V) (US Marine Corps, 2017). Demand quantity x_i is assumed i.i.d Poisson distributed random variable with parameter μ_i. Where, $\mu_i = Vg(i)$, (i.e., the average demand is proportional to the expected total prestage quantity V based on the popularity measure).

For our model, $CV(i)$ is calculated for each SKU using (14). In the CNV problem (we use the superscript " ^ " to denote all parameters and variables of the CNV problem), the critical value is a constant value for all SKUs. Let $\hat{c}_2$ and $\hat{c}_1$ denote the unit cost values in stage 1 and 2, respectively and $\hat{Q}_i^*$ be the optimal recommended prestage quantity for SKU of index i in the CNV problem, calculated by $\hat{Q}_i^* = \min\{\hat{Q}_i \in W | F_i(\hat{Q}_i) \geq \widehat{CV}(i)\}$, where $W = \{0,1,2,3,\ldots\}$, and $\hat{Q}_i$ is the prestaging quantity of SKU i in the CNV problem. The constant critical value for the CNV model is calculated by $\widehat{CV} = \frac{\hat{c}_2 - \hat{c}_1}{\hat{c}_2 - v}$ where $\hat{c}_2$ and $\hat{c}_1$ are calculated by the weighted average method (i.e., $\hat{c}_1 = \sum_{i=1}^{N} C_i^1 g(i)$ and $\hat{c}_2 = \sum_{i=1}^{N} C_i^2 g(i)$). The prestaging quantities for the CNV model ($\hat{Q}_i^*$), and our model (Q_i^*) are calculated using (19). Total prestaging quantities $\sum_{i=1}^{N} Q_i^*$, and $\sum_{i=1}^{N} \hat{Q}_i^*$ of four different cases are shown in Table 6. The cost savings S is defined as the difference in the objective functions (7) of the two models (i.e., $S = Z(Q,x) - \hat{Z}(\hat{Q},x)$). The percent improvement is also defined as $\frac{S}{\hat{Z}(\hat{Q},x)}$. We calculate the average cost savings $\bar{S}$ with standard deviations σ of each model, as well as the average percent improvement (shown as % in Table 6) by randomly generating 100 simulated demand instances. All data analysis and computations for the outlined solution approach are performed on a quad-core 2.7 GHz with 16GB RAM using R 3.3.1 programming language. This included (1) determining the popularity measures for each holds and estimating the γ values based on historical data, (2) simulating instances of random demand, and (3) calculating the inverse Poison CDF value. The analytical model's computation time is negligible.

Given the results from the two models, this numerical example illustrates that the recommendations of our model outperforms the recommendations of the CNV model, but improvements vary by case. Specifically, our model that considers varying unit retrieval costs in the holds results in different policies for prestaging in each of the holds.

Table 6. Estimated parameters and recommended prestaging quantities

Case	*Hold*	*N*	*M*	*V*	*u*	*l*	*r*	*k*	γ	$\sum_{i=1}^{N} Q_i^*$	$\sum_{i=1}^{N} \hat{Q}_i^*$	$\bar{S}$	σ	%
1	13DX	48	938	165	2.1	4.1	2.1	2	1.5378	78	88	30.3	26.5	1.07
2	1TTX	18	1046	200	2.2	2.8	3.4	2	1.7335	18	110	171	70.4	5.78
3	14DX	42	548	110	2.2	2	2.1	4.2	1.8517	0	0	0	0	0
4	2TTX	18	736	140	2.1	4.1	2	2	1.6976	77	81	2.3	3.5	2.36

The two policies recommend different solutions in cases 1, 2, and 4. The biggest difference between recommended prestaging quantities using the two models occurs in case 2. In this case, $\theta = 2.305$, which results from applying Equation (17). Therefore, CSP = [3,18], which means that the first 2 SKUs (high-demanded ones) are not considered for prestaging (i.e., $Q_i^* = 0, \forall i \leq 2$). The policy recommendation of the CNV model is to consider all high, mid, and low-demanded SKUs for the proactive strategy with $\sum_{i=1}^{N} \hat{Q}_i^* = 110$ units of pallets in total. Whereas, the optimal policy recommendation of our model recommends only mid and low-demanded SKUs for the proactive strategy with $\sum_{i=1}^{N} Q_i^* = 18$ pallet units in total, with 5.78 percent improvement on average. The improvement happens for every receiving vehicle and the collective impact on the responsiveness of the military operations is substantial in the long run.

This example highlights that high-demanded SKUs – with index i less than θ – are not always good candidates for prestaging when a set of SKUs with varying costs exist. Instead, mid and low-demanded SKUs warrant a proactive strategy. This phenomena is captured by our model, while the CNV model fails to capture it. In the CNV model, due to the constant costs, once the marginal shortage cost becomes negative, it remains negative. The prestaging starts by considering high-demanded SKUs. When high-demanded SKUs are not considered due to negative marginal shortage costs, other classes will not be considered as well. However, in our model, we consider varying costs due to the different conditions in the holds, and show that marginal shortage costs can have negative values for the high-demanded SKUs, but positive values for mid and low-demanded SKUs. The reason is high-demanded SKUs are already stored in the most convenient locations, making it possible to quickly retrieve them when demand occurs. Thus, these SKUs do not justify the additional double handling required in a proactive strategy and instead should be handled reactively. In cases 1 and 4, both models consider all SKUs for prestaging (i.e., CSP = I^c). However, the optimal solution of our model slightly outperforms the CNV model by recommending lower total prestaging quantities. The reason is that the critical value for high-demanded SKUs are in fact lower than the constant value captured by CNV model (see Figure 6b, and 6c). In case 3, both models perform the same, because they both recommend to prestage none of the SKUs.

7. Conclusion and extensions

Motivated by a sea-based logistics application, we explicitly model the internal cargo flow process of a ship as a floating distribution center. We study the proactive and reactive strategies of a set of SKUs stored in the holds of the ship. Our model captures two unique features: (1) highly dense storage results in SKUs having varying unit fulfillment costs, and (2) time-consuming double handling efforts of prestaging can make a proactive strategy more costly than a reactive strategy. By analyzing the critical value through the lens of activity profiling, insights on whether to handle SKUs with a proactive or reactive strategy are identified.

We extend the two-stage multi-item newsvendor model by considering variable marginal costs for a set of SKUs. Through activity profiling, we define general two-stage cost functions, capable of distributing incurred fulfillment costs proportionally based on the SKU's popularity measure. In doing so, we contribute to the body of newsvendor models by studying the cases where negative marginal shortage costs can happen. We derive a closed-form expression for the critical point to determine a set of SKUs that should be considered as candidates for a proactive strategy. We classify the considered candidate set for a proactive strategy into different cases and derive bounds on the sets. Using our model, we provide counterintuitive insights that shows in particular cases, depending on the application's environment and the relationship between the SKU dependent and independent costs, high-demanded SKUs are better to be excluded from a proactive strategy. Instead, mid and low-demanded SKUs have to be considered. Finally, we illustrate our model with empirical data from a sea-based logistics application.

Extensions to this work include relaxing assumptions to consider systems with more complexities. For example, a more complex demand structure can be explored that represents SKU substitutions, and family of products. This would especially be applicable in proposed seabasing applications that require anticipating the needs of soldiers on the ground. The uncertainty level of SKUs' demand are assumed to be comparable; this can be extended to consider varying demand forecast accuracy based on activity profiles. This work was motivated by an application that had ample resource and storage capacities in both stages; however, an extension would be to consider environments with proactive resource and storage capacities. Moreover, we studied negative marginal shortage cost of the newsvendor with a risk neutral model, which can be extended to risk averse and loss averse newsvendor modeling frameworks.

Acknowledgements
This research is supported by the Office of Naval Research Young Investigator Program: The Design of Responsive Sea-Based Logistic Delivery Systems, Award Number N00014-13-1-0594 and N000141612870.

Appendix A
The derivation of $G(i)$.

As defined in (1), the continuous representation of $g(i) = \omega i^{-\gamma}$. The integration of the number of times SKU i is requested over all SKUs divided by the total numbers of times the ship received a request is equal to one. Thus the integral of $g(i)$ in the domain I^c needs to be equal to one. Since the domain $I^c = (0, N]$, we translate our open set to the closed set $I^c = [\varepsilon, N]$

$$\int_{i=\varepsilon}^{N} g(i) = \omega \int_{i=\varepsilon}^{N} i^{-\gamma} = 1,$$

Hence, solving for ω, we have

$$\omega = \frac{1}{\int_{i=\varepsilon}^{N} i^{-\gamma}} = \frac{1-\gamma}{N^{1-\gamma} - \varepsilon^{1-\gamma}} \tag{21}$$

And the cumulative fraction $G(i)$ is calculated as:

$$G(i) = \frac{i^{1-\gamma} - \varepsilon^{1-\gamma}}{N^{1-\gamma} - \varepsilon^{1-\gamma}} \tag{22}$$

In order to approximate ε, we set $g(i = 1) = G(i = 1)$, which means that the fraction of times the SKU $i = 1$ is requested is equals the cumulative fraction for the first SKU $i = 1$. It is straight forward to show that the solution of $g(1) = G(1)$ occurs when

$$\varepsilon^{1-\gamma} = \gamma \tag{23}$$

Replacing (23) into (21) and (22), we have:

$$\omega = \frac{1-\gamma}{N^{1-\gamma} - \gamma},$$

$$G(i) = \frac{i^{1-\gamma} - \gamma}{N^{1-\gamma} - \gamma}$$

Trivially, $G(\varepsilon) = 0$ and $G(N) = 1$ and $G(i)$ is a continuous strictly increasing function. Therefore, $0 \leq G(i) \leq 1$ for $i \in I^c$.

Appendix B

Proof that expected cost function is convex.

According to Equation (9), the expected cost function for SKU i is as follows.

$$\Pi_i(Q_i) = C_i^1\mu_i + [(C_i^1 - v)E(Q_i - x_i)^+ + (C_i^2 - C_i^1)E(x_i - Q_i)^+]$$

We can write the expected cost as:

$$\Pi_i(Q_i) = constant + h(Q_i)$$

Where

$$H_i(Q_i) = (C_i^1 - v)E(Q_i - x_i)^+ + (C_i^2 - C_i^1)E(Q_i - x_i)^-$$

If we prove $H_i(Q_i)$ is convex, it is equivalent to $\Pi_i(Q_i)$ is convex.

For convenience, let $C_i^2 - C_i^1 = p$ and $C_i^1 - v = q$. Since $C_i^2 > v$, $p + q > 0$ which we use later on in this proof. Also, let $h(y) = qy^+ + py^-$, where $y^+ = \max(y, 0)$ and $y^- = \max(y, 0)$.

h is convex if:

$$h(\lambda y_1 + (1-\lambda)y_2) \leq \lambda h(y_1) + (1-\lambda)h(y_2) \qquad \forall y_1, y_2 \in \text{dom}(h), \forall\lambda \in [0,1]$$

Equivalently,

$$\begin{aligned} &q\max(\lambda y_1 + (1-\lambda)y_2, 0) + p\max(-(\lambda y_1 + (1-\lambda)y_2), 0) \\ &\quad \leq \lambda\big(q\max(y_1, 0) + p\max(-y_1, 0)\big) + (1-\lambda)\big(q\max(y_2, 0) + p\max(-y_2, 0)\big) \end{aligned} \tag{24}$$

1) if $y_1 \geq 0$ and $y_2 \geq 0$:

One can easily verify the left side and the right side of the inequality (24) are equal. Thus inequality (24) holds, and thus h is convex.

2) if $y_1 \leq 0$ and $y_2 \leq 0$:

One can easily verify the left side and the right side of the inequality (24) are equal. Thus inequality (24) holds, and thus h is convex.

3) if $y_1 \geq 0$ and $y_2 \leq 0$:

3-1) Let us assume $\lambda y_1 + (1-\lambda)y_2 \leq 0$:

Then equality () is reduced to:

$$\begin{aligned} -p\lambda y_1 - p(1-\lambda)y_2 & \\ \leq -p\lambda y_1 + q(1-\lambda)y_2 & \\ 0 \leq (p+q)(1-\lambda)y_2 & \qquad (25) \end{aligned}$$

Since $y_2 \geq 0$, $(1-\lambda) \geq 0$, and $\mathrm{p} + q > 0$, the inequality (25) holds. As a result, inequality (24) also holds. Therefore, h is convex.

3-2) if $\lambda y_1 + (1-\lambda)y_2 > 0$:

Then equality () is reduced to:

$$\begin{aligned} q\lambda y_1 + q(1-\lambda)y_2 & \\ \leq -p\lambda y_1 + q(1-\lambda)y_2 & \\ 0 \leq -\lambda(p+q)y_1 & \qquad (26) \end{aligned}$$

Since $y_1 \leq 0$, $(1-\lambda) \geq 0$, and $\mathrm{p} + q > 0$, the inequality (26) holds. As a result, inequality (24) also holds. Therefore, h is convex.

4) if $y_1 \leq 0$ and $y_2 \geq 0$:

Similar to 3, thus h is convex.

Since h is convex, we can also conclude that H is convex since convexity is preserved by linear transformation and by the expectation operator. Therefore, expected cost function $\Pi_i(Q_i)$ is also convex.

End proof.

Appendix C

Proof that the numerator of the critical value is always less than the denominator.

Theorem: For all $i \in I$, the numerator is always less than the denominator of the critical value function, $CV(i)$, defined in (12).

Proof: The critical value is defined in (12); thus, for the numerator to be less than the denominator, (27) should hold $\forall i \in I^c$.

$$C_i^2 - C_i^1 < C_i^2 - v \qquad (25)$$

Since $C_i^1 > v$, (27) holds and we have:

$$C_i^2 - C_i^1 < C_i^2 - v$$

End Proof.

Appendix D

Critical Point θ

We are looking for the critical point $\theta = \{CV(\theta) = 0 | \theta \in I^c\}$. Here we show that if either of the below conditions hold, there exist exactly one critical point θ and it is an interior point in the domain I^c. Otherwise, θ is infeasible.

(1) $\beta_2 > \beta_1$ and $\alpha_2 < \alpha_1$
(2) $\beta_2 < \beta_1$ and $\alpha_2 > \alpha_1$

From (14) we have:

$$CV(i) = \frac{(\beta_2 - \beta_1)i^\gamma + \alpha_2 - \alpha_1}{\beta_2 i^\gamma + \alpha_2 - v}$$

Condition (1) yields to $CV(i = \varepsilon) < 0$ and $CV(i = N) > 0$. Therefore, due to property 1 and the Bolzano's theorem (Apostol, 1974), there is at least one real root in the domain I^c On the other hand, applying this condition into (17), then $\varphi > 0$ and therefore, $CV(i)$ is strictly increasing, therefore, there is at most one real root in the domain I^c . Thus, we can conclude that if the condition (1) holds, then there exists exactly one real root $\theta = \{CV(\theta) = 0 | \theta \in I^c\}$. In order to find such θ, since $\alpha_2 > v$, setting $CV(\theta) = 0$ is equal to solving

$$(\beta_2 - \beta_1)\theta^\gamma + \alpha_2 - \alpha_1 = 0$$

Therefore, if $\beta_2 \neq \beta_1$,

$$\theta = \left(\frac{\alpha_1 - \alpha_2}{\beta_2 - \beta_1}\right)^{\frac{1}{\gamma}} \tag{20}$$

Similarly, condition (2) yield to $CV(i = \varepsilon) > 0$ and $CV(i = N) < 0$. Therefore, due to property 1 and the Bolzano's theorem, there is at least one real root in the domain I^c. On the other hand, applying condition (2) into (17), then $\delta < 0$ and therefore, $CV(i)$ is strictly decreasing, therefore, there is at most one real root in the domain I^c . Thus, we can conclude that if the condition (2) holds, then there exists exactly one real root $\theta = \{CV(\theta) = 0 | \theta \in I^c\}$.

For all other relative values of ($\alpha_1, \alpha_2, \beta_1$ and β_2) except conditions (1) and (2), such θ cannot be found. Thus, either $CV(i) < 0$, $CV(i) = 0$, or $CV_i > 0$ for all $i \in I^c$. It is straight forward for these conditions to show that either θ is undefined or infeasible. i.e., if there exist a real root, then the value of the radicand in (20) becomes negative which is not in the feasible codomain of I^c.

End Proof.

Appendix E

The cost parameters, the sign of the asymptote s, and the critical point θ are required to define the CSP set.

In general, a feasible solution for θ results in CSP being a subset of I^c. Otherwise, either the CSP set and I^ccoincide and all SKUs have the potential to be beneficial if ordered in the stage 1, or CSP $= \emptyset$ and none of the SKUs are considered to be ordered in the stage 1.

Given $\beta_2 > \beta_1$, from (15), this is equivalent to $s > 0$. In this case there exist exactly one feasible θ if $\alpha_2 < \alpha_1$ (see Appendix D). Applying $\beta_2 > \beta_1$ and $\alpha_2 < \alpha_1$ to (16), it is straightforward to show that $\varphi = (\alpha_1 - v)\beta_2 - (\alpha_2 - v)\beta_1$ is strictly positive. Hence, from (18) we can conclude the $CV(i)$ function is strictly increasing. Thus, for $i \leq \theta$, $CV(i) \leq 0$; and, CSP $= (\theta, N]$. Due to (19), it is guaranteed $Q_i^* = 0, \forall i \leq \theta$. Thus, in the optimal solution, SKUs with low indices (high-demanded SKUs) do not have the potential to be considered for prestaging and should be directly transferred once the demand materializes. However, SKUs $i > \theta$ do have the potential to be considered for prestaging. See Figure 6a. However, when $\alpha_1 \geq \alpha_2$, no feasible solution can be found for θ in the domain I^c according to (20). By applying $\beta_2 > \beta_1$ and $\alpha_2 \geq \alpha_1$ conditions to the $CV(i)$ function (14), the critical value becomes strictly positive for all SKUs (i.e., $CV(i) > 0, \forall i \in I^c$). All SKUs have the potential to be considered for prestaging. (i.e., $Q_i^* \geq 0$, $\forall i \in I^c$). According to the definitions of cost functions, this condition is equal to $C_i^2 > C_i^1$ for all SKUs. This is the classical assumption made in the newsvendor problem in which the cost of reactive policy is always higher when the demand materializes. Therefore, we can conclude that CSP $= (0, N] \equiv I^c$. See Figures 6b and 6c.

Given $\beta_2 = \beta_1$, from (15), this is equivalent to $s = 0$. Also, no feasible solution exist θ in the domain I^c according to (20). However, by applying $\alpha_2 > \alpha_1$ conditions to the $CV(i)$ function (14), the critical value becomes strictly positive for all SKUs (i.e., $CV(i) > 0, \forall i \in I^c$). All SKUs have the potential to be considered for prestaging. (i.e., $Q_i^* \geq 0$, $\forall i \in I^c$). This condition is also equal to $C_i^2 > C_i^1$ for all SKUs, according to the definitions of cost functions. This is the classical assumption made in the newsvendor problem in which the cost of reactive policy is always higher when the demand materializes. Therefore, we can conclude that CSP $= (0, N] \equiv I^c$. See Figure 6f. However, when $\alpha_2 \leq \alpha_1$, the value for the $CV(i)$ function (14) is strictly negative. For all SKUs $C_i^2 \leq C_i^1$, since it is always cheaper (when $\alpha_2 < \alpha_1$) or the same cost (when $\alpha_2 = \alpha_1$) to directly transfer SKUs when the demand materializes, proactive policy is never recommended (i.e., $Q_i^* = 0$, $\forall i \in I^c$). Therefore, we conclude that CSP $\equiv \emptyset$. See Figures 6d and 6e.

Given $\beta_2 < \beta_1$, from (15), this is equivalent to $s < 0$. In this case there exist exactly one feasible θ if $\alpha_2 > \alpha_1$ (see Appendix D). Applying $\beta_2 < \beta_1$ and $\alpha_2 > \alpha_1$ to (16) result in $\varphi < 0$. Hence, $CV(i)$ function is strictly decreasing according to (18). Therefore, $CV(i) \leq 0, \forall i \geq \theta$ and CSP $= (0, \theta)$. Due to (19), it is guaranteed $Q_i^* = 0$, $\forall i \geq \theta$. In other words, in the optimal solution, SKUs with higher indices (low-demanded SKUs) than θ are not beneficial to be considered as prestaging, but SKUs with lower indices than θ should be considered for prestaging. See Figure 6i. However, when $\alpha_2 > \alpha_1$, no feasible solution can be found for θ in the domain I^c according to (20). By applying $\beta_2 < \beta_1$ and $\alpha_2 \leq \alpha_1$ conditions to the $CV(i)$ function (14), the critical value becomes strictly negative for all SKUs (i.e., $CV(i) < 0, \forall i \in I^c$). This means that $C_i^2 < C_i^1$ for all SKUs according to the cost functions definitions. Therefore, the decision maker would rather directly transfer all SKUs at the time the demand materializes, resulting in no SKUs being considered for prestaging. Therefore, we conclude that CSP $\equiv \emptyset$. See Figures 6g and 6h.

End Proof.

References

Apostol, T. M. (1974). *Mathematical analysis* (Vol. 2): Addison-Wesley Reading, MA.

Arrow, K. J., Harris, T., & Marschak, J. (1951). Optimal inventory policy. *Econometrica: Journal of the Econometric Society*, 250-272.

Awwad, M., & Pazour, J. (2018). Search Plan for a Single Item in an Inverted T K-deep Storage System. *Military Operations Research.*

Bailey, J. P., & Rabinovich, E. (2005). Internet book retailing and supply chain management: an analytical study of inventory location speculation and postponement. *Transportation Research Part E: Logistics and Transportation Review, 41*(3), 159-177.

Bartholdi, J., & Hackman, S. (2011). Warehouse & Distribution Science, The Supply Chain and Logistics Institute, School of Industrial and Systems Engineering. *Atlanta*.

Beddoes, M. W. (1997). *Logistical implications of operational maneuver from the sea*. Retrieved from

Bender, P. S. (1981). Mathematical modeling of the 20/80 rule: theory and practice. *Journal of Business Logistics, 2*(2), 139-157.

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

Cattani, K. D., Dahan, E., & Schmidt, G. M. (2008). Tailored capacity: speculative and reactive fabrication of fashion goods. *International Journal of Production Economics, 114*(2), 416-430.

Chung, C.-S., & Flynn, J. (2001). A newsboy problem with reactive production. *Computers & Operations Research, 28*(8), 751-765.

Chung, C.-S., Flynn, J., & Kirca, Ö. (2008). A multi-item newsvendor problem with preseason production and capacitated reactive production. *European Journal of Operational Research, 188*(3), 775-792.

Clark, V. (2002). *Sea Power 21: Projecting decisive joint capabilities*. Retrieved from

Combined Joint Operations from the Sea Center of Excellence. (2012). An Introduction to Joint Operations on and from the Sea. Retrieved from http://www.cjoscoe.org/images/12_022-02_Encl1_CJOS_Handbook_Joint_Ops_from_the_Sea.pdf

Eeckhoudt, L., Gollier, C., & Schlesinger, H. (1995). The risk-averse (and prudent) newsboy. *Management Science, 41*(5), 786-794.

Eynan, A., & Rosenblatt, M. J. (1995). Assemble to order and assemble in advance in a single-period stochastic environment. *Naval Research Logistics (NRL), 42*(5), 861-872.

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

Gallego, G., & Moon, I. (1993). The distribution free newsboy problem: review and extensions. *Journal of the Operational Research Society*, 825-834.

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.

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

Hillier, F. S., & Price, C. C. (2012). International Series in Operations Research & Management Science.

Kang, K., & Gue, K. R. (1997). *Sea based logistics: distribution problems for future global contingencies.* Paper presented at the Proceedings of the 29th conference on Winter simulation.

Khouja, M. (1996). A note on the newsboy problem with an emergency supply option. *Journal of the Operational Research Society, 47*(12), 1530-1534.

Khouja, M. (1999). The single-period (news-vendor) problem: literature review and suggestions for future research. *Omega, 27*(5), 537-553.

Kodama, M. (1995). Probabilistic single period inventory model with partial returns and additional orders. *Computers & Industrial Engineering, 29*(1-4), 455-459.

Levin, K. D., & Friedman, Y. (1982). Optimal deployment of logistic units in dynamic combat conditions. *European Journal of Operational Research, 9*(1), 41-46.

Murray Jr, G. R., & Silver, E. A. (1966). A Bayesian analysis of the style goods inventory problem. *Management Science, 12*(11), 785-797.

Parsons, D. (2013). Marine Corps Struggles With Sea-Based Supply Lines. Retrieved from http://www.nationaldefensemagazine.org/archive/2013/march/pages/marinecorpsstrugglesswithsea-basedsupplylines.aspx

Pazour, J. A., & Shin, I. (2016). Logistics Models to Support Order-Fulfillment from the Sea. *Progress in Material Handling Research: 2016*: Material Handling Institute, Charlotte, NC.

Pilnick, S., Glazebrook, K., & Gaver, D. (1991). Optimal sequential replenishment of ships during combat. *Naval Research Logistics (NRL), 38*(5), 637-668.

Prasad, A., Stecke, K. E., & Zhao, X. (2011). Advance selling by a newsvendor retailer. *Production and Operations Management, 20*(1), 129-142.

Qin, Y., Wang, R., Vakharia, A. J., Chen, Y., & Seref, M. M. (2011). The newsvendor problem: Review and directions for future research. *European Journal of Operational Research, 213*(2), 361-374.

Qiu, J., & Sharkey, T. C. (2013). Integrated dynamic single-facility location and inventory planning problems. *IIE Transactions, 45*(8), 883-895.

Reilly, P. J., Pazour, J. A., & Schneider, K. R. (2017). Propagation of unit location uncertainty in dense storage environments. *International Journal of Production Research*, 1-15.

Reimann, M. (2011). Speculative production and anticipative reservation of reactive capacity by a multi-product newsvendor. *European Journal of Operational Research, 211*(1), 35-46.

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.

Scala, N. M., & Pazour, J. A. (2016). A Value Model for Asset Tracking Technology to Support Naval Sea-Based Resupply. *Engineering Management Journal, 28*(2), 120-130.

Silver, E. A., Pyke, D. F., & Peterson, R. (1998). *Inventory management and production planning and scheduling* (Vol. 3): Wiley New York.

Turken, N., Tan, Y., Vakharia, A. J., Wang, L., Wang, R., & Yenipazarli, A. (2012). The multi-product newsvendor problem: Review, extensions, and directions for future research *Handbook of Newsvendor Problems* (pp. 3-39): Springer.

US Marine Corps. (2017). Maritime Expeditionary Warfare Annual Report 2017. Retrieved from http://www.mccdc.marines.mil/Portals/172/Docs/Seabasing/Documents/Annual%20Reports/The%20Maritime%20Expeditionary%20Warfare%20Report%202017.pdf?ver=2017-08-07-144936-783

Wang, C. X., & Webster, S. (2009). The loss-averse newsvendor problem. *Omega, 37*(1), 93-105.

Willey, M. A. (1997). *Demonstrating the Requirements for Amphibious Ready Group (ARG) Replenishment in Sea-Based Logistics Operations*. Retrieved from

Zhang, B., & Du, S. (2010). Multi-product newsboy problem with limited capacity and outsourcing. *European Journal of Operational Research, 202*(1), 107-113.

Chapter 5: Analytical Models for Retrieving and Repositioning Items in Dense Storage Systems and Optimizing the Location of an Open Square

This chapter has been submitted in the following peer-reviewed journal:

1. Climes, Joan, and Pazour, Jennifer A., "Analytical Models for Retrieving and Repositioning Items in Dense Storage Systems and Optimizing the Location of an Open Square," submitted paper.

ANALYTICAL MODELS FOR RETRIEVING AND REPOSITIONING ITEMS IN DENSE STORAGE SYSTEMS AND OPTIMIZING THE LOCATION OF AN OPEN SQUARE

Abstract

Dense storage systems, found in warehousing and sea-based logistics, use space efficiently; however, require repositioning of stored items to retrieve other more densely desired items. We create an optimal procedure that minimizes travel distance required to retrieve and reposition the blocking items in a double inverted T configuration. For varying storage densities and demand profiles, we apply these models and gain new insights on the trade-offs between storage density and travel effort. Repositioning procedures and distance calculations are then updated to determine the value of an empty space, defined by the amount which the system's expected repositioning distance is reduced.

Keywords: Warehousing; Dense Storage; Storage and Retrieval Operations; Travel Time Models; Material Handling

1. Introduction

This work develops analytical models to describe the critical trade-off between increased storage density and response times, measured through travel distance, in dense storage environments. Dense storage systems require the repositioning of items in order to retrieve desired items behind them. Dense storage systems are found frequently in commercial and military ships, where items in the form of packing-crates, cargo, pallet or vehicles are stored densely in the holds of ships. When logistics operations offload all cargo stored at a single destination, the primary metric of interest is to maximize storage density. However, recently the military is interested in sea-based operations, in which ships are used as floating distribution centers, requiring cargo be "selectively offloaded" [1]. This creates a trade-off between storage density and throughput capacities, which are heavily influenced by retrieval efforts to transfer stored cargo on ships [1].

A similar trade-off occurs in last-mile distribution due to the increasing demand for two-hour delivery for online orders in urban environments. To meet this requirement, products are stored near or within the city, which has high storage costs. Dense storage in combination with mobile robotic systems, such as KIVA systems [2], have the potential to balance the need for speed and storage efficiency in urban last mile delivery.

This research focuses on developing analytical models for optimal repositioning and retrieving items in a specific dense storage system, an inverted double T dense storage layout. We are the first to develop analytical models to calculate for every square a retrieval distance, which is the distance it takes to move an item to the exit point, and a repositioning distance, the distance it takes to clear the path for the retrieval space and move the items back to their original locations. Using our analytical models for varying storage densities and demand profiles, we provide insights on the trade-offs between storage density and travel effort. We then determine the best location for an open square that maximizes the reduction in expected repositioning distance of the dense storage system. We identify a benefit tradeoff: the number of retrieval squares that receive this reduction in repositioning distance and the magnitude of the reduction in repositioning distance by which each retrieval square receives.

The remainder of this paper is organized as follows. Section 2 summarizes related research. In Section 3, the system is described, optimal policies are presented, and analytical models are derived for retrieval and repositioning in an inverted double T system. In Section 3, the developed models are used to provide results of travel distance performance for varying storage densities and demand profiles. In Section 4, the analytical models to calculate repositioning distance with the addition of an open square are derived. Results for placement of an open square are presented in Section 5. Finally, our contributions and future research are presented in Section 6.

2. Literature Review

A large body of literature exists on warehouse operations and warehouse design research; see the two papers by Gu, Goetschalckx and McGinnis for detailed reviews [3] [4]. Within their operations framework, this research contributes to research on storage operations, in general, and more specifically, dense storage operations. Therefore, we review the academic literature on dense storage systems, classifying the work into manual and automated dense storage systems.

1.1. Manual Dense Storage Systems

Gue proposes a solution to the layout problem for very high density storage systems of physical goods [5]. These systems are categorized by the possibility of having to move some items out of the way to retrieve the desired item [5]. His objective is to maximize storage density for a given maximum lane depth, k, which allows all items to be retrieved by moving at most $k-1$ items out of the way [5]. Using this maximum lane depth and an $m \times n$ rectangular layout, Gue develops a Fill-and-Rotate algorithm to solve the layout problem. The length and width is recommended to be less than or equal to multiples of $2k+1$ and a bound on storage density of $2k/(2k+1)$ is found to exist [5]. The highest storage density for a fixed value of the accessibility constant k is achieved in layouts that resemble the inverted T configuration [5]. Gue is the seminal paper to consider dense storage and layouts, but does not capture retrieval effort explicitly. Futcher works to facilitate selective offloading operations in dense storage environments by analyzing the tradeoff between retrieval times and storage density using simulation modeling [6]. Futcher concludes that storage densities between 70 and 85% best support selective offload requirements [6]. Futcher's work is similar in that it deals with dense storage and selective offloading. However, Futcher uses simulation modeling. Therefore, this work extends Gue's work by using his layout as a starting point and then developing analytical models to capture retrieval and repositioning performance, as well as procedures to identify the benefit and location of an additional open space.

Due to the repositioning of stored items required when retrieving items in dense storage environments, item location uncertainty can occur [7] [8]. Reilly, Pazour, and Schneider study the case when item location certainty exists initially, followed by subsequent location changes with a lack of asset tracking [8]. Using Markovian principles, Reilly, Pazour, and Schneider model storage systems as finite state Markov chains to estimate the mean search time in steady state given the number of locations that require being searched [8]. Awwad and Pazour study the problem dense storage systems create in terms of location uncertainty and develop procedures for searching an inverted T k-deep storage system for a single item [9] and multiple items [10]. Our research similarly looks at dense storage systems, but assumes repositioned items are returned to their original locations once a desired item is retrieved. Therefore, location uncertainty is not a factor when developing analytical models for retrieval and repositioning nor when analyzing where to locate open spaces.

The literature on warehouse reshuffling or reslotting [11] [12] considers how to reposition items given a current and future storage assignment. Similarly, analytical models rely on a repositioning location, which is an unoccupied square. However, all reslotting research is focused on traditional aisle-based storage systems. In such a system, travel distances between two points follows the rectilinear distance metrics and there is no need to consider repositioning items to gain access to more densely stored items.

1.2. Automated Dense Storage Systems

Gue and Kim propose an algorithm for puzzle-based storage systems – which are automated dense storage systems, in which each cell is able to move up, down, left or right if there is an open cell to do so. They study systems with one open cell, and compare puzzle-based storage systems with aisle-based storage systems [13]. Aisle-based systems are recommended when the objective is to produce the lowest expected retrieval time for a certain number of items, and puzzle-based systems are recommended if the required storage density exceeds 90% [13]. Zaerpour, Yu, and de Koster study live based storage systems [14]. These systems are comprised of units that can move in the x and y directions as in puzzle-based systems, but also have multiple levels of storage [14]. Retrieval times of live-cube storage systems are found to be 30% of traditional systems [14]. Gue, Furmans, Kai, Seibold, and Unldag propose a puzzle-based dense storage system with decentralized control that they call GridStore [15]. Individual conveyers are merged together to automatically retrieve items in a similar way to the 15-puzzle problem [15]. Related is the 15 puzzle problem, see [16] for a review of past work.

Recently, research has studied mobile robotic fulfillment systems, such a KIVA systems, in which robots bring moveable shelves to workstations [2]. Lamballais, Roy, and de Koster use queuing networks to analyze order throughput, order cycle time, and robot utilization with random or class based storage policies [17]. Boysen, Briskorn, and Emde study sequencing of orders and moving racks [18]. Because their focus is pick station operations, their work does not explicitly capture travel patterns of the robots [18]. Bozer and Aldarondo develop a simulation model of a Kiva system and compare the performance to miniload systems [19]. Each of these studies considers mobile robotic fulfillment systems, in which the shelves are not densely stored; however, robotic mobile fulfillment systems are ideal for dense storage systems due to their dynamic shelving design. In fact, current designs resemble the inverted T configuration studied in this work, but typically use multiple I/O points.

Another application of automated dense storage systems occurs at container terminals [20]. Because containers are stacked on top of each other, "rehandles" or "reshuffles" are required to gain access to more deeply stored containers [21] [22]. Container stacking literature considers assignment and sequencing problems that minimize the number of reshuffles required. Also, the impact of temporary storage locations have been studied using simulation to improve operations [23]. Given the material handling equipment used for storage and retrieval, the layout, and operations differ, the developed analytical models are not directly applicable to our work.

1.3. Research Gaps and Contribution

Unlike the work done in puzzle-based, cube-based and gird-store systems, in our research, the aisle locations are static and operations require items to travel along aisles, and the items have fixed locations. Thus, travel time models for automated dense storage systems are not applicable. In the realm of manual storage systems, like Futcher, we are also motivated by selective offloading operations in dense storage, but differ from his work in that we develop analytical models for retrieval and repositioning, as well as consider the addition of an open space to these systems. Our research builds off of Gue's work, using his inverted T layout as a starting layout, however, his work does not capture travel explicitly. Thus, this is the first work to analytically capture retrieval and repositioning performance, as well as to identify and quantify the benefit of adding an open space to the inverted double T layout.

2. Analytical Models for Retrieval and Repositioning Distances in an Inverted Double T System

We build upon Gue's work by using his inverted T layout [5]. This system assumes that all items are stored in equal sized square units. We refer to these equal sized units as squares. A square could be a pallet, a cargo container, or a vehicle, as long as the items stored are of approximately the same size and square. A dense storage system has a maximum lane depth, k, that ensures that all items in these squares will be accessible by moving no more than $k-1$ items out of the way [5]. A system is also given a height variable, h. Occupied squares are squares that contain items for retrieval. Unoccupied squares can occur for one of three reasons: the square belongs to an aisle, the square has been left open, or the square contains an item that was repositioned to gain access to other more deeply-stored items. We restrict retrieval and repositioning travel to unoccupied squares. We make the following assumptions in this work: (1) we consider an inverted double T system configuration, which uses the notation as shown in Figure 2-1; (2) we retrieve and reposition items one-at-a-time to the I/O point (denoted as a start in Figure 3-1); (3) all items are assumed returned to their original locations after use (4) our models capture steady state performance; (5) each item has a given and known demand probability to be retrieved; (6) a single picker conducts all retrieval and repositioning operations; and (7) the picker appears at the location of the retrieval/repositioning square.

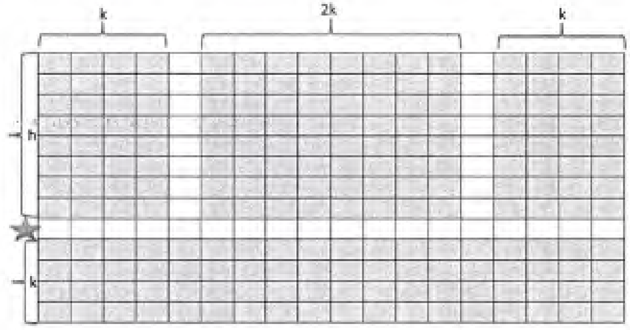


Figure 2-1 Inverted double T layout

Each inverted double T system is therefore $(4k + 2)$ squares by $(k + h + 1)$ squares. Storage density is defined as the ratio of occupied squares to total squares (unoccupied and occupied). The number of occupied squares in an inverted T system is defined as N where $N = k(4k + 2) + 4kh$. Such a configuration achieves high storage density to accessibility, given an upper bound on storage density is $2k/(2k+1)$ [5]. We assign a coordinate system where the origin is at the bottom left-hand corner, and resides outside the system. There is one exit or I/O point with x-coordinate 0 and y-coordinate $k+1$, which we denote as *xExit* and *yExit*, respectively, and is represented by the red star in Figure 2-1. The system has three aisles: one horizontal aisle with y-coordinate $k+1$ and two partial vertical aisles, starting at y-coordinate $k+1$ and stretching until y-coordinate $k+h+1$. One of the vertical aisles is located at x-coordinate $k+1$ and one at x-coordinate $3k+2$. Blocks for the system are defined as shown in Figure 2-2. As summarized in Table 2-1, the blocks and aisles are defined using x- and y-coordinates. While we develop general equations, applicable for any k and h value, we use an inverted double T system with k=4 and h=8 throughout the paper for demonstration purposes.

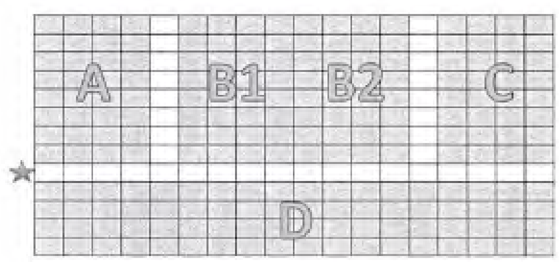


Figure 2-2 Blocks of the inverted double T system

Table 2-1 Block and aisle locations

Block/Aisle	**x-coordinate**	**y-coordinate**
Horizontal Aisle	All	$k+1$
Vertical Aisle 1	$k+1$	$> k+1$
Vertical Aisle 2	$3k+2$	$> k+1$
Block A	$< k+1$	$> k+1$

Block B1	$> k+1$ and $< 2k+2$	$> k+1$
Block B2	$> 2k+1$ and $< 3k+2$	$> k+1$
Block C	$> 3k+2$	$> k+1$
Block D	All	$< k+1$

Because we sometimes have to move items stored in squares out of the way to retrieve items in squares deeper in the system, to understand the throughput requirements in these systems we focus on the properties of two square types. The first we call the *retrieval square*. To obtain systematic results, we evaluate every occupied square in the system as a retrieval square. When accessing an item in a retrieval square requires repositioning of other items, we call these occupied squares in the way repositioning squares, which we abbreviate as *repo squares*. The properties of retrieval and repo squares are outlined in Table 3-2 and 3-3, respectively.

Table 2-2 Properties of the Retrieval Squares

Property	**Definition**
retrievalsquare.x	The x-coordinate of the retrieval square
retrievalsquare.y	The y-coordinate of the retrieval square
retrievalsquare.demandprob	The probability that the item in the retrieval square is requested for retrieval.
retrievalsquare.numbermovedoutofway	The number of items required to be repositioned in order to access the item in the retrieval square
retrievalsquare.retrievaldistance	The distance the item in the retrieval square has to travel to the exit point via the unoccupied squares
retrievalsquare.repositioningdistance	The sum of the repositioning distances of the repositioning squares
retrievalsquare.block	The block that the retrieval square belongs to
retrievalsquare.half	If the square belongs to block D, *retrievalsquare.half=0*. If the square belongs to blocks A or B1, *retrievalsquare.half=1*. If the square belongs to blocks B2 of C, *retrievalsquare.half=2*.
retrievalsquare.direction	The direction the retrieval square will travel toward an aisle
retrievalsquare.i	*k-retrievalsquare.numbermovedoutofway*
retrievalsquare.xLane & retrievalsquare.yLane	The x- and y-coordinates of the nearest aisle position, respectively
retrievalsquare.xXwall & retrievalsquare.yXwall	The x- and y-coordinates of the horizontal wall, respectively. In this case the x-coordinate is $4k+2$ and the y-coordinate is $k+1$

retrievalsquare.xYwall & retrievalsquare.yYwall	The x- and y-coordinates of the nearest vertical wall, respectively. In this case the y-coordinate is $k+h+1$ and the x-coordinate is either $k+1$ or $3k+2$
reposquare.x & reposquare.y for all repo squares	The x- and y-coordinates of each of the repo squares for the items needed to be repositioned in order to access the item in the retrieval square

Table 2-3 Properties of Repositioning Squares

Property	Definition
reposquare.x	The x-coordinate of the repo square
reposquare.y	The y-coordinate of the repo square
reposquare.repositioningdistance	The distance the item in the repo square has to travel to its' repositioning location via the unoccupied squares
reposquare.block	The block that the repo square belongs to
reposquare.direction	The direction the repo square will travel toward an aisle
reposquare.xLane & reposquare.yLane	The x- and y-coordinates of the nearest aisle position, respectively
reposquare.xXwall & reposquare.yXwall	The x- and y-coordinates of the horizontal wall, respectively. In this case the x-coordinate is $4k+2$ and the y-coordinate is $k+1$
reposquare.xYwall & reposquare.yYwall	The x- and y-coordinates of the nearest vertical wall, respectively. In this case the y-coordinate is $k+h+1$ and the x-coordinate is either $k+1$ or $3k+2$
reposquare.repolocx & reposquare.repolocy	The x- and y-coordinates of the location that the item in the repositioning square will be repositioned to

2.1. Optimal Repositioning

First, we describe the structure of the repositioning procedure and develop an algorithm to find the optimal retrieval and repositioning movements. To retrieve any item in a dense storage system, all of the blocking items should be moved first to other unoccupied squares in the system to make a way between the retrieval point and exit point (I/O). The number of the items required to be moved is dependent to the depth of retrieving item. To develop the optimal repositioning square and repositioning algorithm, first, we construct the graph representation of the aisles in a dense storage system, which we call an *Aisle Graph*. As shown in Figure 2-3, to construct such a graph, each unoccupied square in the aisle is represented by a node.

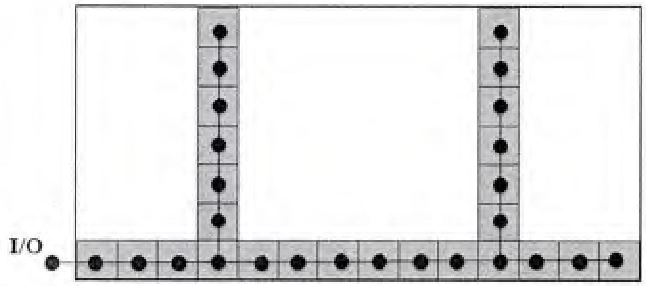


Figure 2-3 An Aisle Graph, which is a graphical representation of the aisles in a dense storage system

Gue (2007) shows that the aisle graph is a spanning tree [13]. One of the basic properties of a spanning tree is that, there exists only one path between any two nodes in the graph. Based on this property, there is only one path between any retrieval square (retrieving node) and the I/O position in a dense storage system. The properties of the aisle structure allows us to develop an optimal procedure to minimize the travel distance required in moving the blocking items. We determine the optimal *Repositioning Squares* locations (blocking the retrieval square) as the closest squares (nodes) from the repositioning square that belong to the set of nodes not on the retrieving path. This is done through a three step process:

STEP 1- Sort the nodes in the non-retrieving path based on their distances to the retrieval square node (in monotonically increasing order).

STEP 2- Select the first $retrievalsquare.numbermovedoutofway$ nodes of the ordered set from STEP 1.

STEP 3- Move the first blocking item to the last node of the ordered set from STEP 2, move the next blocking item to the next last node of the ordered set from STEP 2, and so on.

To illustrate, imagine the dense storage system as shown in Figure 2-4. Suppose an item with $retrievalsquare.numbermovedoutofway = 2$ is to be retrieved from the location shown in Figure 3-4. Step 1 results in the ordered set of squares on the non-retrieving path as $\{G, H, I, F, E, D, C, B, A\}$, ordered based on the distances to the retrieving node (see Section 3.2). Step 2 selects the first 2 nodes from STEP 1's ordered set, resulting in ordered set $\{G, H\}$. Then in Step 3, the first reposquare item should be moved to node G, and the second reposquare item to node H.

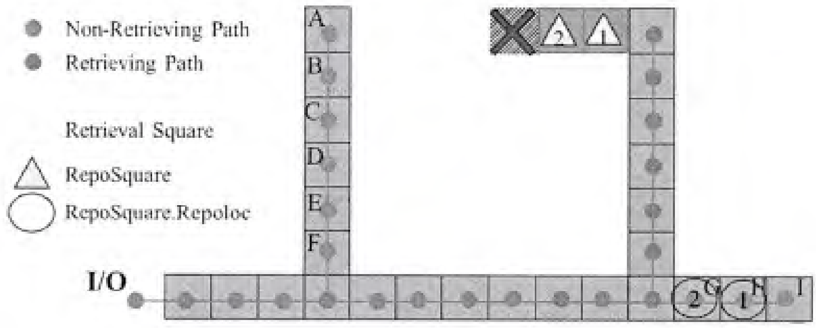


Figure 2-4 Optimal repositioning algorithm for a $retrievalsquare.numbermovedoutofway = 2$

3.2 Retrieval Distance Equations

Retrieval distance assumes an "open" path—thus, total distance for deeply stored items will consist of retrieval and repositioning distances. To calculate the retrieval distance of each retrieval square in the system, the directions that the retrieval squares will travel towards an aisle must be determined. The inverted double T layout is constructed so that retrieving an item would only require repositioning at most *k-1* items. This ensures that an item in a retrieval square will have to travel across at most *k* squares to reach an aisle. The directions the items in the retrieval squares will travel are based on minimizing the distance to the aisle. In block D, the minimum distance to the aisle is always achieved by traveling up to the horizontal aisle. In block A, the item in the retrieval square will travel either down to the horizontal aisle or right to the vertical aisle. If there is a tie in distance to the aisle, the square will travel via the horizontal aisle. The same concept is applied in blocks B1, B2, and C. These results are shown in Figure 2-5.

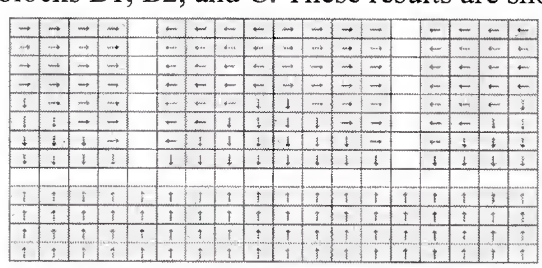


Figure 2-5 Travel Directions of Retrieval Squares for k=4, h=8

To calculate the retrieval distance of each retrieval square, we sum the rectilinear distance between the retrieval square and the nearest aisle square and the rectilinear distance between the aisle square and the exit point. Thus, the equation to calculate the retrieval distance is in Equation (3-26).

$$\begin{aligned} retrievalsquare.retrievaldistance &= |retrievalsquare.x - retrievalsquare.xLane| \\ &+ |retrievalsquare.y - retrievalsquare.yLane| \\ &+ |retrievalsquare.xLane - xExit| \\ &+ |retrievalsquare.yLane - yExit| \end{aligned} \quad (3\text{-}26)$$

When k=4, h=8, the coordinates of the exit point are (0,5). Applying Equation (3-26), the retrieval distances for the k=4, h=8 inverted double T system are shown in Figure 2-6.

17	16	15	14		14	15	16	17	26	25	24	23		23	24	25	26
16	15	14	13		13	14	15	16	25	24	23	22		22	23	24	25
15	14	13	12		12	13	14	15	24	23	22	21		21	22	23	24
14	13	12	11		11	12	13	14	23	22	21	20		20	21	22	23
5	12	11	10		10	11	12	13	14	21	20	19		19	20	21	22
4	5	10	9		9	10	11	12	13	14	19	18		18	19	20	21
3	4	5	8		8	9	10	11	12	13	14	17		17	18	19	20
2	3	4	5		7	8	9	10	11	12	13	14		16	17	18	19
2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22

Figure 2-6 Retrieval Distances for k=4, h=8 inverted double T system

In traditional warehouses, because of the existence of the traditional cross aisles the retrieval travel distance for each storage locations follows the rectilinear distance. However, in the dense storage systems under our study, because of the special structure of the aisles, the retrieval travel distances do not follow the rectilinear distance for all storage locations. As shown in Figure 2-6, for some of the storage locations in Blocks A and B2, *extra* travel distance is required to first arrival at an aisle.

3.2 Repositioning Distance Equations

To calculate the repositioning distance of a retrieval square, we sum the repositioning distances of each of the repositioning squares whose stored items are moved out of the way to gain access to the item in the retrieval square. To do this, we develop analytical expressions to denote the optimal repositioning location, which must be an unoccupied square. The optimal repositioning locations can be determined by the x- and y-coordinates of the repositioning square and the i value of the retrieval square, which represents the depth of a retrieval square, and is calculated using Equation (3-27).

$$retrievalsquare.i = k - retrievalsquare.numbermovedoutofway \qquad (3\text{-}27)$$

The number of items required to be moved out of the way in order to gain access to an item in the retrieval square calculation depends on the direction the item in the retrieval square will travel. If *retrievalsquare.direction* is horizontal, Equation (3-28) is used. If *retrievalsquare.direction* is vertical, Equation (3-29) is used.

$$retrievalsquare.numbermovedoutofway = |retrievalsquare.x - retrievalsquare.xLane| - 1 \qquad (3\text{-}28)$$

$$retrievalsquare.numbermovedoutofway = |retrievalsquare.y - retrievalsquare.yLane| - 1 \qquad (3\text{-}29)$$

Each retrieval square, requires identifying its optimal repositioning squares and their associated repositioning distances. We loop a counter from 1 to *number moved out of way*, and add or subtract the counter value to *retrievalsquare.x* or *retrievalsquare.y* to identify *reposquare.x* and *reposquare.y*. We then calculate the repositioning distances of each repositioning square by identifying their repositioning location. For any i value, we can identify each retrieval square associated with it and their corresponding repositioning squares. For a *k* value of 4, there are 4 possible sets of *i* values: 1, 2, 3, or 4.

We define normal repositioning behavior as a repositioning square traveling towards the aisle and then towards its aisle wall, away from the exit point, so as not to block the path for the item in the retrieval square. We refer to the locations resulting from this "normal" behavior as "good" repositioning locations. The number of "good" repositioning locations is calculated using Equation (3-30) if *retrievalsquare.direction* uses the vertical aisle and Equation (3-31) if *retrievalsquare.direction* uses the horizontal aisle.

$$retrievalsquare.numberofgoodspots = retrievalsquare.yYwall - retrievalsquare.y \tag{3-30}$$

$$retrievalsquare.numberofgoodspots = retrievalsquare.xXwall - retrievalsquare.x \tag{3-31}$$

Unfortunately, not all repositioning squares always all fit in "good" squares without blocking the path for the retrieval item. As such, the item in these repositioning squares are forced to move towards the exit point in their associated nearest aisle until reaching the next aisle, and then moving towards the wall of that aisle. This adds a lot of distance, and we refer to squares requiring this pattern as "bad" squares. Items in repositioning squares can either all travel to "good" squares, all to "bad" squares, or be split: fit as many items as possible into "good" squares with the remainder traveling to "bad squares." Thus, each retrieval square's corresponding repositioning square is given a category: we color each repositioning square gray if all of a retrieval square's corresponding items in repositioning squares can relocate to "good" locations; when the location of the repositioning square requires items traveling to a "bad" location, we color the repositioning square blue; when an item in a repositioning square can fit into a "good" location, but not all items being repositioned can fit into "good" locations, we color the square orange. If an item in an occupied square does not require repositioning because it is deeper than the retrieval square, we color it pink, and if it does not require repositioning due to the direction of travel of surrounding occupied squares, we color it yellow. Retrieval squares are colored green. The resulting categories from *retrievalsquare.i* values of 1, 2, 3 and 4 are shown in Figure 2-7, Figure 2-8, Figure 2-9, and Figure 2-10, respectively.

If *retrievalsquare.numberofgoodspots* ≥ *retrievalsquare.numbermovedoutofway*, all of the retrieval square's associated repositioning squares are colored in gray. If retrievalsquare.numberofgoodspots = 0, then all of the retrieval square's associated repositioning squares are colored blue. When *0 < retrievalsquare.numberofgoodspots < retrievalsquare.numbermovedoutofway*, it results in the split in coloring of repositioning squares between orange and blue. The number of repositioning squares colored orange are *retrievalsquare.numberofgoodspots* and are the ones closest to the aisle, and the blue-colored repositioning squares are closest to the retrieval square.

In Figure 2-7, *retrievalsquare.i* = 1. First, consider retrieval square at *(1,4)*. Its' repositioning squares are located at *(2,9), (3,9,)* and *(4,9)*. Ideally, each repositioning square would travel right towards Vertical Aisle 1 and then up towards the wall. *Retrievalsquare.yYwall*

= *k+1+h*, which is equal to 13 and *retrievalsquare.y = 4*. Because *retrievalsquare.numberofgoodspots = 13-9 =4 > 0*, there is enough room for all of the items in repositioning squares to fit into "good" repositioning locations. The item in repositioning square at *(4,4)* will move up the aisle 3 squares, the item in repositioning square at *(3,4)* will move up the aisle 2 squares, and the item repositioning square at *(2,4)* will move up the aisle 1 square. Second, consider retrieval square at *(1, 13)*. Its' repositioning squares are located at *(2,13)*, *(3,13)*, and *(4,13)*. Given *Retrievalsquare.yYwall = k+1+h* =13 and *retrievalsquare.y = 13, the retrievalsquare.numberofgoodspots = 13-13 =0,* and all of the items in repositioning squares go to "bad" repositioning locations. The item in repositioning square at *(4,13)* will travel over to the vertical aisle, down that aisle, and then towards the horizontal aisle wall 3 squares. The item in repositioning square at *(3,13)* will travel over to the vertical aisle, down that aisle, and then towards the horizontal aisle wall 2 squares. The item in repositioning square at *(2,13)* will travel over to the vertical aisle, down that aisle, and then towards the horizontal aisle wall 1 square. Finally, consider retrieval square at *(1,12)*. Given *Retrievalsquare.yYwall = k+1+h* =13 and *retrievalsquare.y = 12*., then *retrievalsquare.numberofgoodspots = 13-12=1*. Thus, 1 item fits into a "good" repositioning location, and the remaining two items require travel to bad repositioning locations. The item in the repositioning square at *(4,12)* travels to the one "good" repositioning location, up the vertical aisle 1 square. The items in repositioning squares at *(2,12)* and *(3,12)* travel over to the vertical aisle, down that aisle, and then towards the horizontal aisle wall 2 squares and 1 square, respectively.

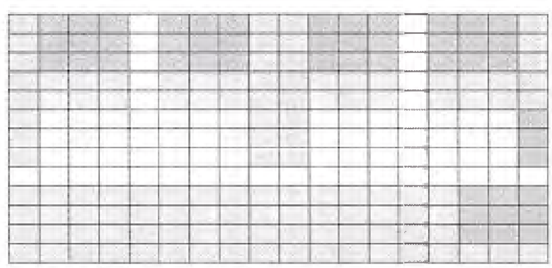


Figure 2-7 Repositioning categories for retrievalsquare.i = 1

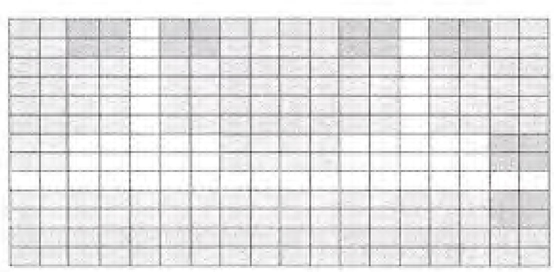


Figure 2-8 Repositioning categories for retrievalsquare.i = 2

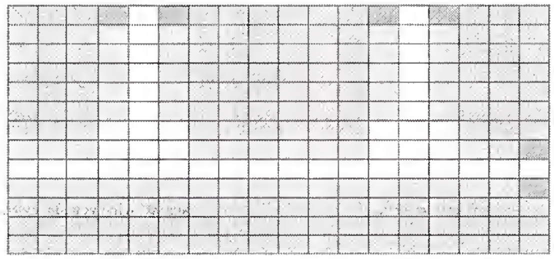


Figure 2-9 Repositioning categories for retrievalsquare.i = 3

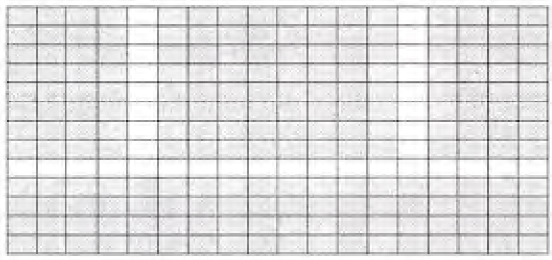


Figure 2-10 Repositioning categories for retrievalsquare.i = 4

For each of the categories of the repositioning squares, we develop equations to calculate the repositioning locations' x and y coordinates. These calculations depend on the block that the repositioning square belongs to, its category, and its direction of travel. The equations for *reposquare.repolocx* and *reposquare.repolocy*, based on these three properties, are found in

Table 2-4.

Table 2-4 Equations for reposquare.repolocx & reposquare.repolocy

Block	Direction	Category	reposquare.repolocx	reposquare.repolocy
D	Horizontal	Gray	*reposquare.x+reposquare.y-retrievalsquare.i*	*k+1*
D	Horizontal	Orange	*reposquare.x+(reposquare.y-k)+(xYwall-reposquare.x)*	*k+1*
D	Horizontal	Blue	*3*k+2*	*k+1+reposquare.y-i*
A	Horizontal	Gray	*reposquare.x+2k-reposquare.y+2-retrievalsquare.i*	*k+1*
A	Vertical	Gray	*k+1*	*reposquare.y+reposquare.x-retrievalsquare.i*
A	Vertical	Orange	*k+1*	*reposquare.y+(reposquare.x-k)+(yXwall-reposquare.y)*
A	Vertical	Blue	*k+1+reposquare.x-retrievalsquare.i*	*k+1*

B1	Horizontal	Gray	*reposquare.repolocx=reposquare.x +2k-reposquare.y+2-retrievalsquare.i*	*k+1*
B1	Vertical	Gray	*k+1*	*reposquare.y+(2*k+1-reposquare.x)-retrievalsquare.i+1*
B1	Vertical	Orange	*k+1*	$reposquare.y + (k - reposquare.x) + (yXwall - reposquare.y) + 2$
B1	Vertical	Blue	*2*k+3-reposquare.x-retrievalsquare.i+k*	*k+1*
B2	Horizontal	Gray	*reposquare.x+2k-reposquare.y+2-retrievalsquare.i*	*k+1*
B2	Vertical	Gray	*3*k+2*	*reposquare.y-2*k-1+reposquare.x-retrievalsquare.i*
B2	Vertical	Orange	*3*k+2*	*reposquare.y+(reposquare.x-3*k-1)+(yXwall-reposquare.y)*
B2	Vertical	Blue	*reposquare.repolocx=k+1+reposquare.x-retrievalsquare.i*	*k+1*
C	Horizontal	Gray	*reposquare.x+2k-reposquare.y+2-retrievalsquare.i*	*k+1*
C	Horizontal	Orange	*reposquare.x+(k+2-reposquare.y)+(xYwall-reposquare.x)*	*k+1*
C	Horizontal	Blue	*3*k+2*	*3*k+3-reposquare.y-retrievalsquare.i*
C	Vertical	Gray	*3*k+2*	*reposquare.y+(4*k+2-reposquare.x)-retrievalsquare.i+*
C	Vertical	Orange	*3*k+2*	*reposquare.y+(3*k-reposquare.x)+(yXwall-reposquare.y)+3*
C	Vertical	Blue	*7*k+5-reposquare.x-retrievalsquare.i*	*k+1*

After computing *reposquare.repolocx* and *reposquare.repolocy*, we calculate *reposquare.repositioningdistance* using Equation (3-32). This equation captures the travel distance over unoccupied squares in aisles and where other items have been repositioned. The distances to the repositioning locations for each item in a repositioning square given *retrievalsquare.i=1, 2, 3, and 4* are shown in Figure 2-11 – Figure 3-14, respectively.

$$\begin{aligned} reposquare.repositioningdistance &= 2 * |reposquare.x - reposquare.xLane| \\ &+ |reposquare.xLane - reposquare.repolocx| \\ &+ |reposquare.y - reposquare.yLane| \\ &+ |reposquare.yLane - reposquare.repolocy| \end{aligned} \tag{3-32}$$

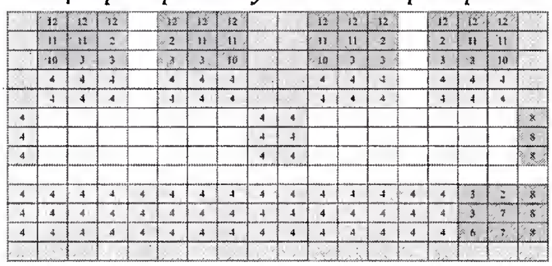


Figure 2-11 Repositioning distances for retrievalsquare.i=1

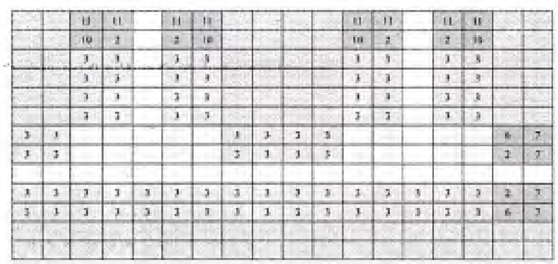


Figure 2-12 Repositioning distances for retrievalsquare.i = 2

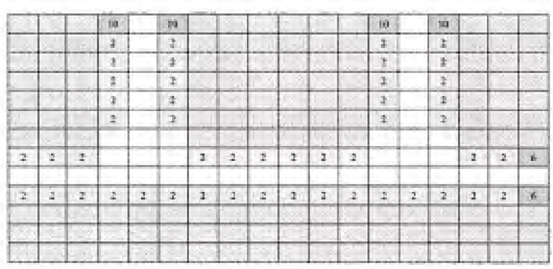


Figure 2-13 Repositioning distances for retrievalsquare.i = 3

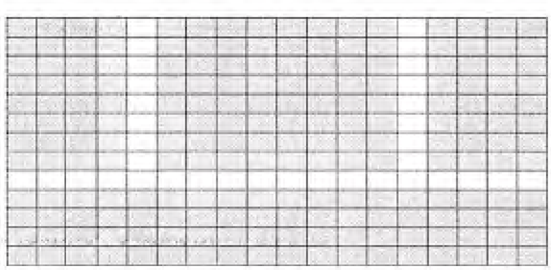


Figure 2-14 Repositioning distances for retrievalsquare.i = 4

The total repositioning distance for the retrieval square is calculated using Equation (3-33), which is the sum of repositioning distances of all the items blocking the retrieval square. The total repositioning distance of the retrieval squares for the example *k=4*, *h=8* system is shown in Figure 2-15.

$$retrievalsquare.repositioningdistance = \sum_{j=1}^{retrievalsquare.numbermovedoutofway} reposquare.repositioningdistance_j \quad (3\text{-}33)$$

72	44	28	0		0	28	44	72	72	44	28	0		0	28	44	72
48	24	4	0		0	4	24	48	48	24	4	0		0	4	24	48
32	12	4	0		0	4	12	32	32	12	4	0		0	4	12	32
24	12	4	0		0	4	12	24	24	12	4	0		0	4	12	24
24	12	4	0		0	4	12	24	24	12	4	0		0	4	12	48
12	12	4	0		0	4	12	12	12	12	4	0		0	4	16	28
4	4	4	0		0	4	4	4	4	4	4	0		0	4	4	12
0	0	0	0		0	0	0	0	0	0	0	0		0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	12
12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	16	28
24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	24	52	48

Figure 2-15 Retrievalsquare.repositioningdistance

3. Results of Travel Distance Performance in Dense Storage Systems

In this section, we use our developed analytical models to provide insights into the trade-off between storage density and travel distances. For each occupied square, the sum of *retrievalsquare.retrievaldistance* and *retrievalsquare.repositioningdistance* is shown in Figure 3-1 for *k=4*. In traditional aisle-based warehouses with *k=1*, the travel distance for each storage location are increasing linearly as approaching the farther storage locations from I/O point. However, as Figure 3-1 shows, some storage locations have total retrieval and repositioning distances relatively large compared to others. These large differences occur for two reasons. First, some squares require *extra retrieval distance* beyond rectilinear travel. Also, for some squares situated at the end of the aisles, to avoid blocking, the optimal repositioning square, and associated repositioning distance, is large.

89	60	35	14		14	35	60	89	98	69	44	23		23	44	69	98
64	39	18	13		13	18	39	64	73	48	27	22		22	27	48	73
47	26	17	12		12	17	26	47	56	35	26	21		21	26	35	56
38	25	16	11		11	16	25	38	47	34	25	20		20	25	34	47
29	24	15	10		10	15	24	37	38	33	24	19		19	24	33	70
16	17	14	9		9	14	23	24	25	26	18	17		17	22	23	32
7	8	9	8		8	13	14	15	16	17	18	17		17	22	23	32
2	3	4	5		7	8	9	10	11	12	13	14		16	17	18	19
2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	32
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	36	49
29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	53	70

Figure 3-1 Retrieval & repositioning distances of retrieval squares with k=4

Next, we compare retrieval and repositioning performance of dense storage systems with varying densities by exploring systems with *k=1, 2, 3,* and *4*. Assuming each occupied square has a known, given demand probability, we calculate the expected total distance of the system using Equation (4-1).

$$expected\ system\ distance = \sum_{j=1}^{N} [retrievalsquare.demandprob_j * (retrievalsquare.retrievaldistance_j + retrievalsquare.repostioningdistance_j)] \quad (4\text{-}1)$$

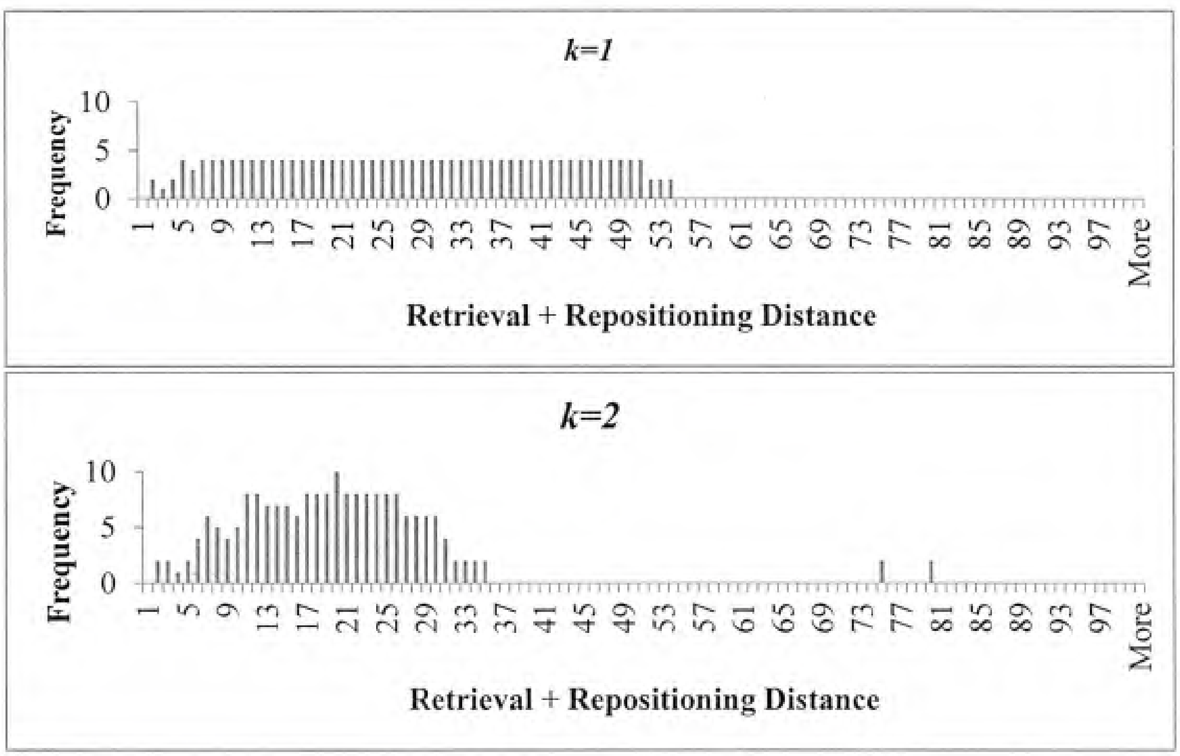
For comparative purposes, we vary *h* so that *N* is approximately equal to 200 (see Table 3-1). Interestingly, *k=2* has the lowest expected performance when each occupied square is equally likely to be requested, i.e., *retrievalsquare.demandprob* is equal to *1/N* for all squares. This occurs because a *k=2* system balances the reduction in y-travel with the increased need for repositioning with deeper systems. This is in contrast to a *k=1* system, which requires a long and skinny (i.e., *h=48*) configuration to achieve *N=198* occupied squares. Therefore, a *k=1* system requires long y-distance travel for retrieval, but no repositioning travel. While a configuration with *k=3* results in less retrieval travel distance in the y-direction than *k=2*, the increase in repositioning travel distance does not offset this reduction. Therefore, if a system needs around *N=200* occupied squares and demand for the items stored are equally likely to be requested, the best configuration is to design a system with *k=2*.

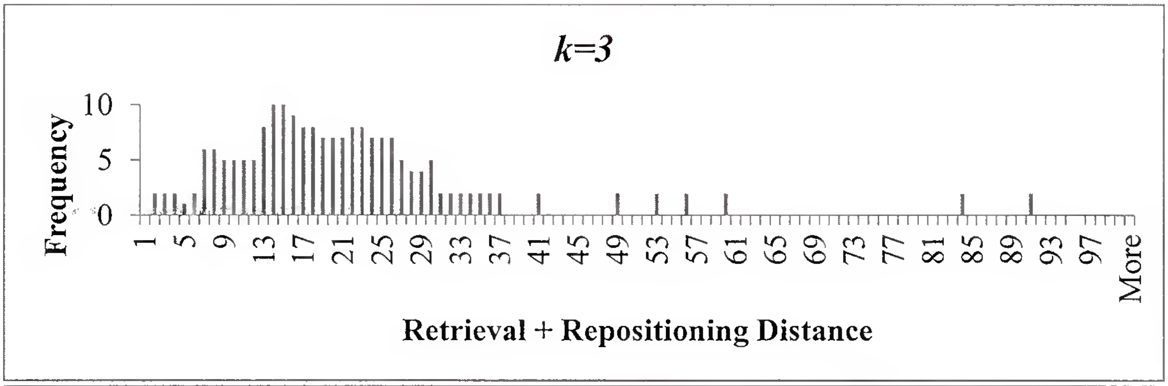
Table 3-1 Expected System Distance for varying densities and demand skewness curves

k	h	N	20/20	20/40	20/60	20/80
1	48	198	28.24	20.51	14.83	9.64
2	22	196	20.08	15.62	11.83	8.22
3	13	198	21.79	16.20	12.18	8.41
4	8	200	26.04	18.16	13.27	8.90

However, as displayed in the histograms in Figure 3-2, the retrieval plus repositioning distance of each occupied square varies greatly. The standard deviation of retrieval plus repositioning times in systems with *k=1* is 14.39, *k=2* is 11.35, with *k =3* is 14.26, and with *k=4* is 18.03. Therefore, the second moment performance results in *k=2* also achieving the lowest standard deviation, but then the next lowest standard deviation is *k=3,* followed by *k=1,* and then *k=4.* Such variability in square performance is due to some squares near the I/O point being very convenient, while others – at the top corners requiring both long y-distance travel, as well as long repositioning distances. This variability in square performance indicates that dedicated assignment policies, which assign the most convenient locations to the most active items, could improve expected system performance if the demand probability is not equally likely for all items. Therefore, we explore expected system performance when we assign the most demanded SKUs to the squares with the smallest retrieval plus repositioning distances.

To explore different skewness levels over the N items, we model the ABC curve as presented by Bender [24]. This allows us to vary demand probabilities, such that a few stored items make up a large total percent of demand requests. Specifically, we explore a X/Y curve, where X% of items make of Y% of demand. For $k=1$ to 4, Table 4-1 presents expected system distance results for 20/40, 20/60, and 20/80 curves, which corresponds to S=0.60, S=0.20, and S=0.0667, respectively in Bender (1981). These results are compared to items that are equally likely to be demanded, which is equivalent to a 20/20 curve. To determine the probability an item was requested, we assumed 200 items in the Bender model. For storage systems with less than 200 occupied squares, it was assumed that the least convenient square stored all remaining $200-N$ items. As the ABC curve becomes more skewed, the expected system distance decreases; this occurs across all values of k. The value of k that minimizes the expected system distance is $k=2$ for all demand curves tested. This analysis leads us to recommend systems with $k=2$ to be preferred over systems with $k=1$, as a system with $k=2$ has reduced expected travel distance (and thus increased expected throughput in the system), as well as increased density. However, for $k>3$, a trade-off exists. Increasing k, decreases the expected throughput in the system, but increases the storage density of the system (which increases as k increase, and relates to the storage efficiency).





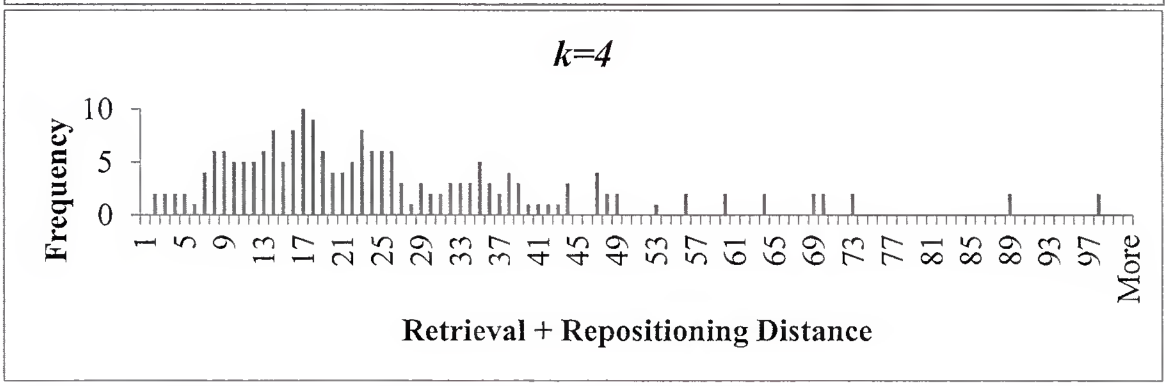


Figure 3-2 Histograms of the retrieval plus repositioning distances of all N *occupied squares with* k=1, 2, 3, and 4.

4. Adjustment of Analytical Models to Incorporate Properties of an Open Square

In this section we consider the inverted T configuration and the option to introduce an open square. This analysis is motivated by discussions with Naval and Marine Corps personnel, who wanted to know where space should be provided for repositioning activity that is most helpful in a dense storage environments. Therefore, we explore the placement of an additional open square, placed in a square that would have typically stored an item, but is now left open. The properties of this open square are defined in Table 4-1. Although an open square is unoccupied, its location is that of a square that would have been occupied in our original model. We use this fact to calculate properties for an imaginary "item" in the open square that ease calculating the benefit of the open square.

Table 4-1 Properties of an Open Square

Property	Definition
opensquare.x	The x-coordinate of the open square
opensquare.y	The y-coordinate of the open square
opensquare.depth	The distance from the open square to its nearest aisle location
opensquare.block	The block that the open square belongs to

opensquare.half	If the square belongs to block D, *opensquare.half=0*. If the square belongs to blocks A or B1, *opensquare.half=1*. If the square belongs to blocks B2 of C, *opensquare.half=2*.
opensquare.direction	The direction the open square will travel toward an aisle
opensquare.i	*k-opensquare.depth*
opensquare.xLane & *opensquare.yLane*	The x- and y-coordinates of the nearest aisle position, respectively
opensquare.xXwall & *opensquare.yXwall*	The x- and y-coordinates of the horizontal wall, respectively. In this case the x-coordinate is $4k+2$ and the y-coordinate is $k+1$
opensquare.xYwall & *opensquare.yYwall*	The x- and y-coordinates of the nearest vertical wall, respectively. In this case the y-coordinate is $k+h+1$ and the x-coordinate is either $k+1$ or $3k+2$
opensquare.aislex &*opensquare.aisley*	The x- and y-coordinates of the nearest square that borders an aisle to the open square. These squares have an i value of k
opensquare.systemdistance	The total distance of the system given each item in a retrieval square is requested for retrieval individually and the presence of the open square

The introduction of an open square in a system does not affect the retrieval distance of any of the items in the system, but it can impact the repositioning distances in one of three ways: (1) the number of items required to move out of the way can decrease by one, (2) the repositioning location of one of the repositioning squares may change to the open square, if the distance to the open square is closer than that of its original repositioning location, or (3) it may not impact the repositioning distance at all. If it does impact the repositioning distance, the repositioning locations of each repositioning square need updated.

To calculate the total system distance given the introduction of an open square, we first calculate *opensquare.i*, *opensquare.numbermovedoutofway*, *opensquare.ailsex*, and *opensquare.aisley*. The i value of the open square is representative of the depth of an open square, and is calculated using Equation (5-1). The depth of the open square depends on the *opensquare.direction*. If *opensquare.direction* is horizontal, Equation (5-2) is used. If *opensquare.direction* is vertical, Equation (5-3) is used.

$$opensquare.i = k - opensquare.depth \tag{5-1}$$

$$opensquare.numbermovedoutofway = |opensquare.x - opensquare.xLane| - 1 \tag{5-2}$$

$$opensquare.numbermovedoutofway = |opensquare.y - opensquare.yLane| - 1 \tag{5-3}$$

The calculations for *opensquare.ailsex*, and *opensquare.aisley* depend on *opensquare.block*, and *opensquare.direction*. Table 4-2 summarizes the values of *opensquare.ailsex*, and *opensquare.aisley*.

Table 4-2 opensquare.aislex & opensquare.aisley values

Block	Direction	*opensquare.aislex*	opensquare.aisley
D	Horizontal	*opensquare.x*	*k*
A	Vertical	*opensquare.y*	*k*
A	Horizontal	*opensquare.x*	*k+2*
B1	Vertical	*opensquare.y*	*k+2*
B1	Horizontal	*opensquare.x*	*k+2*
B2	Vertical	*opensquare.y*	*3*k+1*
B2	Horizontal	*opensquare.x*	*k+2*
C	Vertical	*opensquare.y*	*3*k+1*
C	Horizontal	*opensquare.x*	*k+2*

5.1 Updated Retrieval & Repositioning Square Properties

We consider that any occupied square can be designated as an open square. Given that open square, we evaluate each remaining occupied square as a retrieval square. Each retrieval square has the same properties of the base case, with the additional properties outlined in Table 4-3.

Table 4-3 Additional Properties of a retrieval square given an open square

Property	Definition
retrievalsquare.numbermovedchange	If the number of items moved out of the way required to reach the retrieval item is decreased, *retrievalsquare.numbermovedchange=1*. Otherwise, *retrievalsquare.numbermovedchange=0.*
retrievalsquare.locationchange	If *retrievalsquare.locationtest* is less than or equal to the original repositioning distance of the repositioning square closest to the retrieval square, *retrievalsquare.locationchange=1*. Otherwise *retrievalsquare.locationchange=0.*
retrievalsquare.locationtest	The repositioning distance of the repositioning square closest to the retrieval square to the square that border an aisle closest to the repositioning squares, plus the distance required to move the items closest to the aisle deeper, so that the space closest to the aisle is left open. This value is compared to the original repositioning distance of the repositioning square closest to the retrieval square to determine whether *retrievalsquare.locationchange* will equal 1 or 0.

As mentioned previously, the retrieval distances of the retrieval squares are unaffected by the introduction of the open square. The repositioning distances, however, change if *retrievalsquare.numbermovedchange=1* or *retrievalsquare.locationchange=1*. If this is the case, the repositioning distances of the repositioning squares may change. As with the retrieval squares, the introduction of an open square requires the introduction of additional properties defined in Table 3-4.

Table 4-4 Additional repositioning square properties

Property	Definition
reposquare.depth	The distance from the repo square to its nearest aisle location
reposquare.i	*k-reposquare.depth*
reposquare.openforwardx & *reposquare.openforwardy*	The x- and y-coordinates of the repositioning square one square less deep than the original repositioning square
reposquare.openbackx & *reposquare.openbacky*	The x- and y-coordinates of the repositioning square one square deeper than the original repositioning square
reposquare.originalx & *reposquare.originaly*	Used to keep track of the original repositioning square coordinates

For *retrievalsquare.numbermovedchange* to evaluate to 1, all of the following conditions must hold: (1) *opensquare.block* = *retrievalsquare.block*; (2) *opensquare.direction* = *retrievalsquare.direction*; (3) if *opensquare.direction* is to use the vertical aisle, *retrievalsquare.x* = *opensquare.x*, or if *opensquare.direction* is to use the horizontal aisle, *retrievalsquare.y* = *opensquare.y;* and (4) *retrievalsquare.i* ≤ *opensquare.i.*

To test for the *retrievalsquare.locationchange*, we calculate *retrievalsquare.locationtest* for the repositioning square closest to the retrieval square using Equation (5-4). To use this equation, the open square and the retrieval square must use the same set of aisles (horizontal aisle and vertical aisle 1 or horizontal aisle and vertical aisle 2). For example, a repositioning square corresponding to a retrieval square in block A will not use an open square in Block C as the optimal repositioning location, but a repositioning square in Block D may use an open square in Block A as its optimal repositioning location. To ensure this, we use the *opensquare.half* and *retrievalsquare.half* properties. If *opensquare.half* + *retreivalsquare.half* ≤ *2* or *opensquare.half* = *retrievalsquare.half*, they use the same set of aisles. An open square in Block D can be matched with a retrieval square in any other block and vice versa, but retrieval or open squares in Blocks A and B1 cannot be matched with retrieval or open squares in Blocks B2 and Block C. So, if *opensquare.half* + *retreivalsquare.half* ≤ *2* or *opensquare.half* = *retrievalsquare.half* and the value of *retrievalsquare.locationtest* is less than the original repositioning distance for the repositioning square closest to the retrieval square, *retrievalsquare.locationchange=1*.

$$retrievalsquare.locationtest = (2 * |reposquare.x - reposquare.xLane| + |reposquare.xLane - opensquare.aislex| + |reposquare.y - reposquare.yLane| + |reposquare.xLane - opensquareaisley|) + (2 * (k - opensquare.i)) \quad (5\text{-}4)$$

If the number of items moved out of the way for a given item in a retrieval square changes, there cannot also be a repositioning location change for one of retrieval squares' corresponding items in repositioning squares from its original location to the open square. This is because when the number of items moved out of the way changes, it implies that the open square is in the path needed to travel for the retrieval square to reach the aisle on its way to the exit point. Therefore, if the item in the repositioning square closest to the retrieval square was repositioned to (*opensquare.openaislex*, *opensquare.openaisley*), the path would be blocked, making it not possible to retrieve the item in retrieval square. As such, if *retrievalsquare.numbermovedchange=1* or all conditions (1), (2), and (3) for *retrievalsquare.numbermovedchange=1* are met, we set *retrievalsquare.locationchange=0.*

5.2 Updated Repositioning Equations

If either *retrievalsquare.numbermovedchange=1* or *retrievalsquare.locationchange=1*, the retrieval squares' corresponding repositioning squares will have changes in repositioning distances. There are two obvious changes: (1) when *retrievalsquare.numbermovedchange=1*, the repositioning distance of one of the original repositioning squares without the addition of the open square will be 0 because now there is no item in the open square to move; and (2) when *retrievalsquare.locationchange=1*, the repositioning square closest to the retrieval square will have an updated repositioning location to the open square, decreasing its repositioning distance by one. Others repositioning squares may not have to travel as far because there are less items to move out of the way as to not block the path of the item of the retrieval square.

For example, in our *k=4*, *h=8* system, take a retrieval square at *(1,9)*. The items in repositioning squares located at *(2,9)*, *(3,9)*, and *(4,9)* are repositioned to *(5,10)*, *(5,11)*, and *(5,12)*, respectively. These original repositioning squares and repositioning locations are depicted in Figure 4-1. Now consider the same retrieval square, but with the addition of an open square located at *(2,9)*. Now the items in the repositioning squares are at *(3,9)* and *(4,9)*. These items now only have to travel to *(5,10)* and *(5,11)*, respectively. Notice that the items in *(3,9)* and *(4,9)* when there is an open square present travel to the repositioning location that the items in *(2,9)* and *(3,9)* traveled to when there was no open square in the system. These updated repositioning locations are depicted in Figure 4-2.

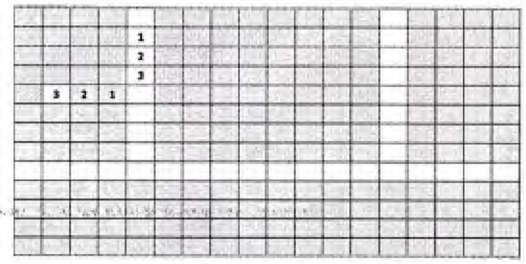


Figure 4-1 Original repositioning locations for retrieval square at (1,9)

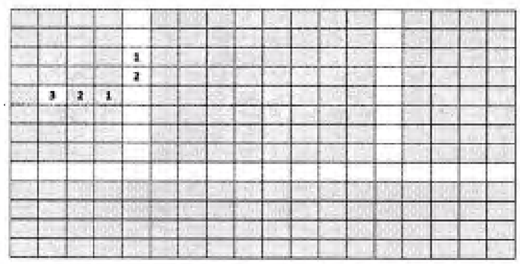


Figure 4-2 Updated repositioning locations for retrieval square at (1,9) and open square at (2,9

Consider the same *k=4*, *h=8* system with a retrieval square at *(1,11)*. The items in repositioning squares are located at *(2,11)*, *(3,11)*, and *(4,11)*, and are repositioned to *(6,5)*, *(5,12)*, and *(5,13)*, respectively. These repositioning squares and repositioning locations are depicted in Figure 4-3. Now consider the same retrieval square in the same system with an open square at *(4,11)*. The items in the repositioning squares are located at *(2,11)* and *(3,11)*, and they now are repositioned to *(5,12)* and *(5,13)*. Notice that the items in *(2,11)* and *(3,11)* when there is an open square in the system travel to the repositioning locations that the items in *(3,11)* and *(4,11)* traveled to when there was no open square in the system. These updated repositioning locations are depicted in Figure 4-4.

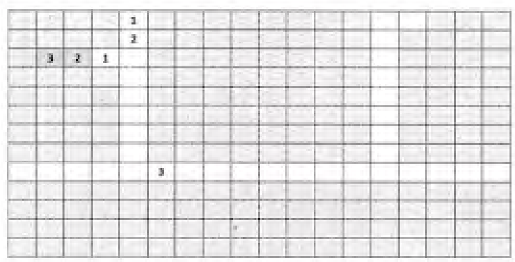


Figure 4-3 Original repositioning locations for retrieval square at (1,11)

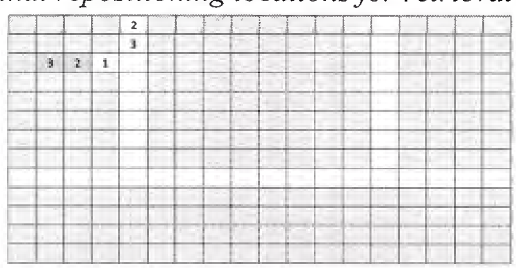


Figure 4-4 Updated repositioning locations for retrieval square at (1,11) and open square at (4,11)

As discussed when describing the system with no open square, we loop over all the repositioning squares and calculate their repositioning locations and repositioning distances individually, and then sum them up to calculate the repositioning distance of the retrieval square. As illustrated in our examples, it is useful to know the coordinates of the repositioning squares surrounding the current repositioning square. We can substitute the coordinates of these squares in for the coordinates of the repositioning square to calculate the repositioning location if certain conditions are met. For each repositioning square, we calculate *reposquare.forwardx*, *reposquare.forwardy*, *reposquare.backx*, and *reposquare.back.* These calculations depend on the block and direction of the repositioning square, and are summarized in

Table 4-5. In addition to calculating these values, we also set *reposquare.orginalx* and *reposquare.originaly* to *reposquare.x* and *reposquare.y*, respectively. We calculate these values regardless if there is a change due to the presence of an open square or not.

Table 4-5 Calculations for reposquare.forwardx &reposquare.forwardy and reposquare.backx & reposquare.backy

Block	**Direction**	**reposquare.forwardx**	**reposquare.forwardy**	**reposquare.backx**	**reposquare.backy**
D	Horizontal	*reposquare.x*	*repossquare.y+1*	*reposquare.x*	*repossquare.y-1*
A	Vertical	*reposquare.x+1*	*repossquare.y*	*reposquare.x-1*	*repossquare.y*
A	Horizontal	*reposquare.x*	*repossquare.y-1*	*reposquare.x*	*repossquare.y+1*
B1	Vertical	*reposquare.x-1*	*repossquare.y*	*reposquare.x+1*	*repossquare.y*
B1	Horizontal	*reposquare.x*	*repossquare.y-1*	*reposquare.x*	*repossquare.y+1*
B2	Vertical	*reposquare.x+1*	*repossquare.y*	*reposquare.x-1*	*repossquare.y*
B2	Horizontal	*reposquare.x*	*repossquare.y-1*	*reposquare.x*	*repossquare.y+1*
C	Vertical	*reposquare.x-1*	*repossquare.y*	*reposquare.x+1*	*repossquare.y*
C	Horizontal	*reposquare.x*	*repossquare.y-1*	*reposquare.x*	*repossquare.y+1*

To determine whether or not a change in the repositioning locations occurs, we need to calculate *reposquare.depth*, which depends on the *reposquare.direction*. If *reposquare.direction* is horizontal, Equation (5-5) is used. If *reposquare.direction* is vertical, Equation (5-6) is used.

$$reposquare.numbermovedoutofway = |reposquare.x - reposquare.xLane| - 1 \quad (5\text{-}5)$$

$$reposquare.numbermovedoutofway = |reposquare.y - reposquare.yLane| - 1 \quad (5\text{-}6)$$

We also use *retrievalsquare.numberofgoodspots* to determine whether or not there will be a change in repositioning locations of repositioning squares. There are four cases of the values *retrievalsquare.numberofgoodspots* that we consider: (1) *retrievalsquare.numberofgoodspots=0*, (2) *retrievalsquare.numberofgoodspots = (retrievalsquare.numbermovedoutofway-1)*, (3) *retrievalsquare.numberofgoodspots > (retrievalsquare.numbermovedoutofway-1)*, and (4) *0 < retrievalsquare.numberofgoodspots < (retrievalsquare.numbermovedoutofway -1)*. The following section considers the first three cases only.

5.2.1 Repositioning locations for retrievalsquare.numberofgoodspots cases 1, 2, and 3

Consider when *retrievalsquare.numbermovedchange=1*. When *reposquare.depth < opensquare.depth* and *retrievalsquare.numberofgoodspots=0*, we update *reposquare.x* and *reposquare.y* to be *reposquare.backx* and *reposquare.backy*, respectively. When *reposquare.depth < opensquare.depth* and *retrievalsquare.numberofgoodspots > (retrievalsquare.numbermovedoutofway-1)*, we update *reposquare.x* and *reposquare.y* to be *reposquare.backx* and *reposquare.backy*, respectively. When *reposquare.depth >*

opensquare.depth and *retrievalsquare.numberofgoodspots = (retrievalsquare.numbermovedoutofway-1)*, we update *reposquare.x* and *reposquare.y* to be *reposquare.forwardx* and *reposquare.forwardy*, respectively. Otherwise, the values of reposquare.x and reposquare.y are unchanged.

Now consider when *retrievalsquare.locationchange=1*. Instead of using the depth of the open square for comparison with the depth of the repositioning square, we use the depth of the repositioning square closest to the retrieval square, since this is where the change in how far the repositioning squares have to travel comes from. The depth of the repositioning square closest to the retrieval square is equal to *k - (retrievalquare.i+1)*. When *reposquare.depth < [k - (retrievalquare.i+1)]* and *retrievalsquare.numberofgoodspots=0*, we update *reposquare.x* and *reposquare.y* to be *reposquare.backx* and *reposquare.backy*, respectively. When *reposquare.depth < [k - (retrievalquare.i+1)]* and *retrievalsquare.numberofgoodspots > (retrievalsquare.numbermovedoutofway-1)*, we update *reposquare.x* and *reposquare.y* to be *reposquare.backx* and *reposquare.backy*, respectively. When *reposquare.depth > [k - (retrievalquare.i+1)]* and *retrievalsquare.numberofgoodspots = (retrievalsquare.numbermovedoutofway-1)*, we update *reposquare.x* and *reposquare.y* to be *reposquare.forwardx* and *reposquare.forwardy*, respectively. Otherwise, the values of reposquare.x and reposquare.y are unchanged.

The repositioning locations are then calculated the same way they were for the system with no open squares using the updated values of *reposquare.x* and *reposquare.y*.

5.2.2 Repositoning locations for retrievalsquare.numberofgoodspots Case 4

Now consider the case when *0 < retrievalsquare.numberofgoodspots < (retrievalsquare.numbermovedoutofway -1)*. This occurs when there are not enough good spots for all the items in the repositioning squares to travel to "good" locations, but there is at least one "good" location for an item to be repositioned. How do the repositioning locations of the repositioning items change when *retrievalsquare.numbermovedchange=1* or *retrievalsquare.locationchange=1*? Unlike in Cases 1, 2 and 3, we cannot update *reposquare.x* and *reposquare.y* to the coordinates of surrounding repositioning squares. Instead, we formulate a new method and equations for calculating the repositioning locations of the items repositioning squares.

Prior to looping over all the repositioning squares, we define two counter variables, *numberlefttomove* and *numberofgoodspotsleft*, and initialize them at *(retrievalsquare.numbermovedoutofway-1)* and *retrievalsquare.numberofgoodspots*, respectively. We loop from the repositioning square closest to the aisle to the repositioning square closest to the retrieval square. We only use the method when the repositioning square is not the open square if *retrievalsquare.numbermovedchange=1* or the repositioning square is not the repositioning square closest to the retrieval square if *retrievalsquare.locationchange=1*.

For each repositioning square, we first check if *numberofgoodspotsleft >0*. If this is true and *reposquare.direction* is the horizontal aisle, Equation (5-7) and Equation (5-8) are used to calculate *reposquare.repolocx* and *reposquare.repolocy*, respectively. If this is true and *reposquare.direction* is the vertical aisle, Equation (5-9) and Equation (5-10) are used to calculate *reposquare.repolocx* and *reposquare.repolocy*, respectively.

$$reposquare.repolocx = retrievalsquare.x + numberofgoodspotsleft \quad (5\text{-}7)$$

$$reposquare.repolocy = retrievalsquare.yLane \quad (5\text{-}8)$$

$$reposquare.repolocx = retrievalsquare.xLane \quad (5\text{-}9)$$

$$reposquare.repolocy = retrievalsquare.y + numberofgoodspotsleft \quad (5\text{-}10)$$

For each case, we update *numberlefttomove* and *numberofgoodspotsleft to (numberlefttomove-1)* and *(numberofgoodspotsleft-1)*, respectively after calculating *reposquare.repolocx* and *reposquare.repolocy*.

What if *numberofgoodspotsleft=0*? Now the repositioning location equations depend on both the direction and block. The calculations for *reposquare.repolocx* and *reposquare.repolocy* are summarized in Table 4-6. For each case, we update *numberlefttomove* to *(numberlefttomove-1)* after calculating *reposquare.repolocx* and *reposquare.repolocy*.

Table 4-6 reposquare.repolocx and reposquare.repolocy equations when numberofgoodpotsleft=0

Block	**Direction**	**reposquare.repolocx**	**reposquare.repolocy**
D	Horizontal	*3*k+2*	*retrievalsquare.yLane+numberlefttomove*
A	Vertical	*retrievalsquare.xLane+numberlefttomove*	*k+1*
A	Horizontal	*k+1*	*retrievalsquare.yLane+numberlefttomove*
B1	Vertical	*retrievalsquare.xLane+numberlefttomove*	*k+1*
B1	Horizontal	*k+1*	*retrievalsquare.yLane+numberlefttomove*
B2	Vertical	*retrievalsquare.xLane+numberlefttomove*	*k+1*
B2	Horizontal	*3*k+2*	*retrievalsquare.yLane+numberlefttomove*
C	Vertical	*retrievalsquare.xLane+numberlefttomove*	*k+1*
C	Horizontal	*3*k+2*	*retrievalsquare.yLane+numberlefttomove*

5.2.3 Updated Repositioning Distance Equations

Because we updated the coordinates of some of the repositioning squares, we have to alter the equations for the repositioning distance of the repositioning squares. If *retrievalsquare.numbermovedchange=1*, we set *reposquare.repositioningdistance* equal to 0 if the open square coordinates are equal to the repositioning square original coordinates. If *retrievalsquare.locationchange=1*, we use Equation (5-11) to calculate *reposquare.repositoningdistance* if the repositioning square is the one closest to the retrieval square. Otherwise, we use Equation (5-12) to calculate *reposquare.repostioningdistance.*

$$reposquare.repositioningdistance = 2(|reposquare.originalx - retrievalsquare.xLane| + |retrievalsquare.xLane - opensquare.aislex| + |reposquare.originaly - retrievalsquare.yLane| + |retrievalsquare.yLane - opensquare.aisley|) + 2 * opensquare.depth \quad (5\text{-}11)$$

$$reposquare.repositoningdistance = 2 * |reposquare.originalx - retrievalsquare.xLane| + |retrievalsquare.xLane - reposquare.repolocx| + |reposquare.originaly - retrievalsquare.yLane| + |retrievalsquare.yLane - reposquare.repolocy| \quad (5\text{-}12)$$

The repositioning distance of the retrieval square is calculated by summing over all repositioning squares using Equation **Error! Reference source not found.**. The expected system distance given the presence of the open square is calculated using Equation **Error! Reference source not found.**. If the coordinates of the retrieval square are equal to the coordinates of the open square, we set *retrievalsquare.demandprob* to 0. Otherwise, we set *retrievalsquare.demandprob* to *[1/(N-1)]*.

5. Results of Open Square Placement

The benefit of an open square placed where a retrieval square was previously located is calculated by subtracting the expected system distance with the open square from the expected system distance without the open square. However, we have to modify the expected system distance of the system without the open square. Otherwise, we would unfairly choose squares that have large repositioning and retrieval distances to be the open square, as these distances go away when we calculate the expected system distance. We define a new property of the open square, *opensquare.baseexpectedsystemdistance*, which we calculate the same way as *opensquare.expectedsystemdistance*, but use the equations for the system without the open square as the retrieval and repositioning distances of each square instead of the equations for the retrieval and repositioning distance equations for the system with the open square. This way, when we compare *opensquare.baseexpectedsystemdistance* with *opensquare.expecteddistance*, we are capturing only the added benefit of the addition of the open square to the system. This is the affect the open square has on the other retrieval squares in the system.

We will use four measures (calculated for each open square location) when determining the best location of the open square in the system: (1) *benefit*, which is the reduction in expected system travel distance of the base case and the expected system travel distance (with the open square) calculated using Equation (6-1); (2) number of squares affected by a change in the number moved out of the way, which we will call *# moved out of way affected* and is calculating using Equation (6-2); (3) number of squares affected by a change in the repositioning location of the closest repositioning square, which we will call *# location affected* and is calculated using Equation (6-3); and (4) total number of squares affected by the presence of an open square, which we call *# affected* and is calculated using Equation (6-4) as the sum of retrieval squares affected either by fewer moved out of the way or by reduced repositioning locations.

$$benefit = opensquare.baseexpectedsystemdistance - opensquare.expectedsystemdistance \quad (6\text{-}1)$$

$$\#\, number\ moved\ out\ of\ way\ affected = \sum_{j=1}^{N} retrievalsquare.numbermovedchange_j \quad (6\text{-}2)$$

$$\#\, location\ affected = \sum_{j=1}^{N} retrievalsquare.locationchange_j \quad (6\text{-}3)$$

$$\#\, affected = \sum_{j=1}^{N} (retrievalsquare.numbermovedchange_j + retrievalsquare.locationchange_j) \quad (6\text{-}4)$$

Determining the best location of the open square in a given system is an optimization problem. The decision variables of the model are the x- and y-coordinates of the open square. We consider two separate objectives, which are either to maximize *benefit* or to maximize *# affected.* The model is constrained by the dynamics of retrieval and repositioning movement in the dense storage environment. These constraints are nonlinear due to the presence of absolute values, which is what makes this problem hard. When we are interested in the single best location for an open square, we use a brute force approach to solve the optimization problem. To do so, we calculate the objective (either the *benefit* or *# affected*) for each occupied square and select the square with the largest objective function value.

What happens for our example system with *k=4*, *h=8*? If we assume equal probability of an item in a retrieval square being selected for retrieval, the benefit of an open square being introduced in this system is shown in Figure 5-1 for the *# number affected*, Figure 5-2 for the *# moved out of the way affected*, Figure 5-3 for the *# location affected*, and Figure 5-4 for the *# affected.*

0.342	0.573	0.774	0.945		0.945	0.774	0.573	0.342	0.342	0.573	0.774	0.945		0.945	0.774	0.573	0.342
0.432	0.653	0.844	0.945		0.945	0.844	0.653	0.432	0.432	0.653	0.844	0.945		0.945	0.844	0.653	0.432
0.412	0.623	0.764	0.874		0.874	0.764	0.623	0.412	0.412	0.623	0.764	0.874		0.874	0.764	0.623	0.412
0.322	0.503	0.653	0.774		0.774	0.653	0.503	0.322	0.322	0.503	0.653	0.774		0.774	0.653	0.503	0.322
0.000	0.322	0.482	0.613		0.613	0.482	0.322	0.000	0.000	0.322	0.482	0.613		0.613	0.482	0.322	0.060
0.060	0.000	0.322	0.462		0.462	0.322	0.000	0.060	0.060	0.000	0.322	0.462		0.462	0.322	0.151	0.191
0.090	0.040	0.000	0.322		0.322	0.000	0.040	0.090	0.090	0.040	0.000	0.322		0.322	0.171	0.291	0.332
0.090	0.050	0.020	0.201		0.201	0.020	0.050	0.090	0.090	0.050	0.020	0.201		0.372	0.291	0.392	0.432
0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.261	0.342	0.412	0.432
0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.090	0.251	0.332	0.332
0.060	0.060	0.060	0.060	0.060	0.060	0.060	0.060	0.060	0.060	0.060	0.060	0.060	0.060	0.060	0.060	0.211	0.191
0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.060

Figure 5-1 Benefit of an open square for an open square in a k=4, h=8 system

0	1	2	3	3	2	1	0	0	1	2	3	3	2	1	0
0	1	2	3	3	2	1	0	0	1	2	3	3	2	1	0
0	1	2	3	3	2	1	0	0	1	2	3	3	2	1	0
0	1	2	3	3	2	1	0	0	1	2	3	3	2	1	0
0	0	1	2	2	1	0	0	0	0	1	2	2	1	0	0
1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1
2	1	0	0	0	0	1	2	2	1	0	0	0	0	1	2
3	2	1	0	0	1	2	3	3	2	1	0	0	1	2	3

3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 5-2 # moved out of the way affected for an open square in a k=4, h=8 system

9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9
10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
11	11	11	11	11	11	11	11	11	11	11	11	11	11	11	11
12	12	12	12	12	12	12	12	12	12	12	12	12	12	12	12
0	12	12	12	12	12	12	0	0	12	12	12	12	12	12	3
0	0	12	12	12	12	0	0	0	0	12	12	12	12	8	3
0	0	0	12	12	0	0	0	0	0	0	12	12	10	8	6
0	0	0	12	12	0	0	0	0	0	0	12	22	10	9	6

0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	9	8	6
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	7	6
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3

Figure 5-3 # location affected for an open square in a k=4, h=8 system

9	10	11	12	12	11	10	9	9	10	11	12	12	11	10	9
10	11	12	13	13	12	11	10	10	11	12	13	13	12	11	10
11	12	13	14	14	13	12	11	11	12	13	14	14	13	12	11
12	13	14	15	15	14	13	12	12	13	14	15	15	14	13	12
0	12	13	14	14	13	12	0	0	12	13	14	14	13	12	3
1	0	12	13	13	12	0	1	1	0	12	13	13	12	8	4
2	1	0	12	12	0	1	2	2	1	0	12	12	10	9	8
3	2	1	12	12	1	2	3	3	2	1	12	22	11	11	9

3	3	3	3	3	3	3	3	3	3	3	3	3	3	13	12	11	9
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	11	9	8
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	8	4
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3

Figure 5-4 # affected for an open square in a k=4, h=8 system

From these results we make the following observations. First, we observe that for a given x- or y-coordinate and direction of travel, it is better to locate the open square along an aisle as opposed to deeper in the system. This makes sense: a retrieval square gains location benefit from being able to move the item in its closest repositioning square to the aisle location of the open square. If the open square is not already along an aisle, a *(2*opensquare.depth)* term exists in the repositioning distance equation, which represents moving items in the path of the open square deeper to vacate the aisle spot. If the open square is along the aisle, we do not incur this distance, resulting in a greater magnitude of benefit. The # *location affected* is the same for a given x- or y-coordinate and corresponding direction of travel. The quantity of retrieval squares affected by a location change is the same, but the magnitude of benefit is impacted by decreasing the *(2*opensquare.depth)* as an open square gets closer to an aisle. Another reason for the greater benefit for open squares along aisles for a given x- or y-coordinate is that the # *number affected* increases as an open square moves towards being located along an aisle. This is because a change in the number of repositioning squares a retrieval square has to move out of the way only occurs when the open square is in front of the retrieval square and there are items that require repositioning. So, as an the open square location moves closer to an aisle, more retrieval squares gain benefit from an open square closer to an aisle, because they become deeper than the open square.

Another observation is that open squares located closer to the walls of the system provide more benefit than those closer to center intersections of the horizontal and vertical aisles. This result also makes sense: if a repositioning square once had to travel to a "bad" location, but now can fit in a "good" location or in the open square aisle location, it greatly reduces the repositioning distance requires to gain access to an item in a retrieval square. Even if the presence of an open square does not produce one of these large benefits, items in repositioning squares can benefit from not having to travel as far to be repositioned if their retrieval square is affected by the open square. However, it is the reduction in the number of times required to reposition to "bad" locations that drives open squares close to walls, to provide the greatest benefit. In this example, locating an open square along an aisle closer to a vertical aisle provides greater impact than locating an open square along an aisle closer to the horizontal wall. This makes sense because in this example, $h > k$.

Another interesting observation is that the benefit of open squares is symmetric for open squares with vertical direction from over the vertical aisles. This makes sense because repositioning distance of each retrieval square was symmetric for our base case. Thus, no square is deemed as the best for an entire system, but provides benefit for a set of squares. For the $k=4$, $h=8$ system, the best locations to place one open square to maximize benefit would be located at $(k, k+h+1)$, $(k, k+h)$, $(k+2, k+1+h)$, $(k+2, k+h)$, $(3k+1, k+h+1)$, $(3k+1, k+h)$, $(3k+3, k+h+1$,, or $(3k+3, k+h)$. This means there are multiple optimal solutions to our problem.

What happens if we vary h? Our $k=4$, $h=8$ example is for the case where $h=2k$. What if $h=k$? The results for the *benefit*, *# moved out of way affected*, *# location affected*, and *# affected* for an open square in a $k=4$, $h=8$ system are shown in Figure 5-5, Figure 5-6, Figure 5-7, and Figure 5-8, respectively.

0.000	0.059	0.207	0.311		0.311	0.207	0.059	0.000	0.000	0.059	0.207	0.311		0.311	0.207	0.059	0.044
0.089	0.000	0.119	0.207		0.207	0.119	0.000	0.089	0.089	0.000	0.119	0.207		0.207	0.119	0.133	0.237
0.133	0.059	0.000	0.119		0.119	0.000	0.059	0.133	0.133	0.059	0.000	0.119		0.119	0.163	0.341	0.444
0.133	0.074	0.030	0.059		0.059	0.030	0.074	0.133	0.133	0.074	0.030	0.059		0.222	0.341	0.489	0.593
0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.296	0.415	0.519	0.593
0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.133	0.281	0.400	0.444
0.089	0.089	0.089	0.089	0.089	0.089	0.089	0.089	0.089	0.089	0.089	0.089	0.089	0.089	0.089	0.089	0.222	0.237
0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.044

Figure 5-5 Benefit for an open square for an open square in a k=4, h=4 system

0	0	1	2		2	1	0	0	0	0	1	2		2	1	0	0
1	0	0	1		1	0	0	1	1	0	0	1		1	0	0	1
2	1	0	0		0	0	1	2	2	1	0	0		0	0	1	2
3	2	1	0		0	1	2	3	3	2	1	0		0	1	2	3
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 5-6 # number moved out of the way affected for an open square in a k=4, h=4 system

0	2	2	2	2	2	2	0	0	2	2	2			2	2	2	3
0	0	4	4	4	4	0	0	0	0	4	4			4	4	8	3
0	0	0	4	4	0	0	0	0	0	0	4			4	10	8	6
0	0	0	4	4	0	0	0	0	0	0	4			14	10	9	6
0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	9	8	6
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	7	6
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3

Figure 5-7 # location affected for an open square in a k=4, h=4 system

0	2	3	4	4	3	2	0	0	2	3	4			4	3	2	3
1	0	4	5	5	4	0	1	1	0	4	5			5	4	8	4
2	1	0	4	4	0	1	2	2	1	0	4			4	10	9	8
3	2	1	4	4	1	2	3	3	2	1	4			14	11	11	9
3	3	3	3	3	3	3	3	3	3	3	3	3	3	13	12	11	9
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	11	9	8
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	8	4
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3

Figure 5-8 # affected for an open square in a k=4, h=4 system

We observe similar results from the *h=k* case. The observation that the squares along the aisle outperform the deeper squares in the system holds. Additionally, we observe that performance of an open square improves as it nears a wall of the system. For the *h=k* case, being located against the horizontal wall is a better location. Again, because of symmetry, there are multiple optimal locations at *(4k+2, k)* and *(4k+2, k+h+1)*.

What if *h=4k*? Now we consider a *k=4, h=16* system. The results for the *benefit*, *# moved out of way affected*, *# location affected*, and *# affected* for an open square in a *k=4, h=16* system are shown in Figure 5-9, Figure 5-10, Figure 5-11, and Figure 5-12, respectively.

0.648	0.838	1.009	1.162	1.162	1.009	0.838	0.648	0.648	0.838	1.009	1.162			1.162	1.009	0.838	0.648
0.752	0.936	1.101	1.162	1.162	1.101	0.936	0.752	0.752	0.936	1.101	1.162			1.162	1.101	0.936	0.752
0.789	0.966	1.052	1.119	1.119	1.052	0.966	0.789	0.789	0.966	1.052	1.119			1.119	1.052	0.966	0.789
0.783	0.893	0.985	1.058	1.058	0.985	0.893	0.783	0.783	0.893	0.985	1.058			1.058	0.985	0.893	0.783
0.709	0.820	0.911	0.985	0.985	0.911	0.820	0.709	0.709	0.820	0.911	0.985			0.985	0.911	0.820	0.709
0.636	0.746	0.838	0.911	0.911	0.838	0.746	0.636	0.636	0.746	0.838	0.911			0.911	0.838	0.746	0.636
0.563	0.673	0.765	0.838	0.838	0.765	0.673	0.563	0.563	0.673	0.765	0.838			0.838	0.765	0.673	0.563
0.489	0.599	0.691	0.765	0.765	0.691	0.599	0.489	0.489	0.599	0.691	0.765			0.765	0.691	0.599	0.489
0.416	0.526	0.618	0.691	0.691	0.618	0.526	0.416	0.416	0.526	0.618	0.691			0.691	0.618	0.526	0.416
0.343	0.453	0.544	0.618	0.618	0.544	0.453	0.343	0.343	0.453	0.544	0.618			0.618	0.544	0.453	0.343
0.269	0.379	0.471	0.544	0.544	0.471	0.379	0.269	0.269	0.379	0.471	0.544			0.544	0.471	0.379	0.269
0.196	0.306	0.398	0.471	0.471	0.398	0.306	0.196	0.196	0.306	0.398	0.471			0.471	0.398	0.306	0.196
0.000	0.196	0.294	0.373	0.373	0.294	0.196	0.000	0.000	0.196	0.294	0.373			0.373	0.294	0.196	0.037
0.037	0.000	0.196	0.281	0.281	0.196	0.000	0.037	0.037	0.000	0.196	0.281			0.281	0.196	0.092	0.116
0.055	0.024	0.000	0.196	0.196	0.000	0.024	0.055	0.055	0.024	0.000	0.196			0.196	0.104	0.177	0.202
0.055	0.031	0.012	0.122	0.122	0.012	0.031	0.055	0.055	0.031	0.012	0.122			0.226	0.177	0.239	0.263
0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.159	0.208	0.251	0.263
0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.055	0.153	0.202	0.202
0.037	0.037	0.037	0.037	0.037	0.037	0.037	0.037	0.037	0.037	0.037	0.037	0.037	0.037	0.037	0.037	0.128	0.116
0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.037

Figure 5-9 Benefit for an open square for an open square in a k=4, h=16 system

0	1	2	3	3	2	1	0	0	1	2	3			3	2	1	0
0	1	2	3	3	2	1	0	0	1	2	3			3	2	1	0
0	1	2	3	3	2	1	0	0	1	2	3			3	2	1	0
0	1	2	3	3	2	1	0	0	1	2	3			3	2	1	0
0	1	2	3	3	2	1	0	0	1	2	3			3	2	1	0
0	1	2	3	3	2	1	0	0	1	2	3			3	2	1	0
0	1	2	3	3	2	1	0	0	1	2	3			3	2	1	0
0	1	2	3	3	2	1	0	0	1	2	3			3	2	1	0
0	1	2	3	3	2	1	0	0	1	2	3			3	2	1	0
0	1	2	3	3	2	1	0	0	1	2	3			3	2	1	0
0	1	2	3	3	2	1	0	0	1	2	3			3	2	1	0
0	1	2	3	3	2	1	0	0	1	2	3			3	2	1	0
0	0	1	2	2	1	0	0	0	0	1	2			2	1	0	0
1	0	0	1	1	0	0	1	1	0	0	1			1	0	0	1
2	1	0	0	0	0	1	2	2	1	0	0			0	0	1	2
3	2	1	0	0	1	2	3	3	2	1	0			0	1	2	3
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 5-10 # number moved out of the way affected for an open square in a k=4, h=16 system

9	9	9	9		9	9	9	9	9	9	9	9		9	9	9	9
10	10	10	10		10	10	10	10	10	10	10	10		10	10	10	10
11	11	11	11		11	11	11	11	11	11	11	11		11	11	11	11
12	12	12	12		12	12	12	12	12	12	12	12		12	12	12	12
12	12	12	12		12	12	12	12	12	12	12	12		12	12	12	12
12	12	12	12		12	12	12	12	12	12	12	12		12	12	12	12
12	12	12	12		12	12	12	12	12	12	12	12		12	12	12	12
12	12	12	12		12	12	12	12	12	12	12	12		12	12	12	12
12	12	12	12		12	12	12	12	12	12	12	12		12	12	12	12
12	12	12	12		12	12	12	12	12	12	12	12		12	12	12	12
12	12	12	12		12	12	12	12	12	12	12	12		12	12	12	12
12	12	12	12		12	12	12	12	12	12	12	12		12	12	12	12
0	12	12	12		12	12	12	0	0	12	12	12		12	12	12	3
0	0	12	12		12	12	0	0	0	0	12	12		12	12	8	3
0	0	0	12		12	0	0	0	0	0	0	12		12	10	8	6
0	0	0	12		12	0	0	0	0	0	0	12		12	10	9	6
0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	9	8	6
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	7	6
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3

Figure 5-11 # location affected for an open square in a k=4, h=16 system

9	10	11	12		12	11	10	9	9	10	11	12		12	11	10	9
10	11	12	13		13	12	11	10	10	11	12	13		13	12	11	10
11	12	13	14		14	13	12	11	11	12	13	14		14	13	12	11
12	13	14	15		15	14	13	12	12	13	14	15		15	14	13	12
12	13	14	15		15	14	13	12	12	13	14	15		15	14	13	12
12	13	14	15		15	14	13	12	12	13	14	15		15	14	13	12
12	13	14	15		15	14	13	12	12	13	14	15		15	14	13	12
12	13	14	15		15	14	13	12	12	13	14	15		15	14	13	12
12	13	14	15		15	14	13	12	12	13	14	15		15	14	13	12
12	13	14	15		15	14	13	12	12	13	14	15		15	14	13	12
12	13	14	15		15	14	13	12	12	13	14	15		15	14	13	12
12	13	14	15		15	14	13	12	12	13	14	15		15	14	13	12
0	12	13	14		14	13	12	0	0	12	13	14		14	13	12	3
1	0	12	13		13	12	0	1	1	0	12	13		13	12	8	4
2	1	0	12		12	0	1	2	2	1	0	12		12	10	9	8
3	2	1	12		12	1	2	3	3	2	1	12		22	11	11	9
3	3	3	3	3	3	3	3	3	3	3	3	3	3	13	12	11	9
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	11	9	8
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	8	4
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3

Figure 5-12 # affected for an open square in a k=4, h=16 system

Again, aisle locations are better than more deeply-placed open squares, and it is better to position open squares along the walls of the system. Because $h>k$ in this system, it is better to place open squares along the vertical walls. Symmetry causes multiple optimal solutions at $(k, k+h+1)$, $(k, k+h)$, $(k+2, k+1+h)$, $(k+2, k+h)$, $(3k+1, k+h+1)$, $(3k+1, k+h)$, $(3k+3, k+h+1$, or $(3k+3, k+h)$.

7.0 Conclusions and Future Research Directions

In summary, this work makes the following contributions. We are the first to derive the optimal repositioning and retrieval paths and associated distance equations for every square in an inverted double T system. This requires deriving expressions for the repositioning locations of the items in repositioning squares, which depend on an item's block, direction of travel, and repositioning location, which are all analytically derived. Second, we determine the best location to place an additional open square that maximizes the expected reduction in repositioning distance in the system. This requires updating the repositioning locations and repositioning expressions for the introduction of an open square. New insights are provided for the trade-off between storage density and retrieval and repositioning effort, as well as placement of open squares. Specifically, systems with an accessibility factor of $k=2$ resulted in the lowest expected performance and are recommended if a system needs around N=200 occupied locations. Such a system is recommended for a wide range of probability skewness due to its able to balance the reduction in y-travel with the increased need for repositioning with deeper systems. In systems with equal probability of item demand, the best locations for an open square occur for squares along the aisle, and close to the vertical walls if $h > k$, or close to the horizontal wall if $h = k$. Due to the symmetry in repositioning distances, multiple optimal solutions exist.

A number of interesting areas for future research exists. Future research could create dynamic policies for nondepleting systems, which require updating layouts for different values of N and multiple open squares. To more accurately study mobile robotic fulfillment system applications, multiple I/O points, corresponding to multiple pick positions and replenishments stations could be explored. Finally, this work proposes steady-state results; transient performance could be explored in the future.

Acknowledgements:
This research has been partly supported by the Office of Naval Research Young Investigator Program: The Design of Responsive Sea-Based Logistic Delivery Systems, Award Number N00014-13-1-0594 and N000141612870. We also acknowledge Faraz Ramtin for developing the spanning tree results.

References

[1] J. A. Pazour and I. Shin, "Logistics Models to Support Order-fulfillment from the Sea," in *Progress in Material Handling Research*, Charlotte, MHI, 2016.

[2] J. Enright and P. R. Wurman, "Optimization and Coordinated Autonomy in Mobile Fulfillment Systems," in *Automated action planning for autonomous mobile robots*, 2011.

[3] J. Gu, M. Goetschalckx and L. F. McGinnis, "Research on warehouse operation: A comprehensive review," *European Journal of Operational Research,* vol. 177, no. 1, pp. 1-21, 2007.

[4] J. Gu, M. Goetschalckx and L. F. McGinnis, "Research on warehouse design and performance evaluation: A comprehensive review," *European Journal of Operational Research,* vol. 203, no. 3, pp. 539-549, 2010.

[5] K. R. Gue, "Very high density storage systems," *IIE Transactions,* vol. 38, no. 1, pp. 79-90, 2006.

[6] F. Futcher, "Selective offload capability simulation (SOCS): an analysis of high-density storage configurations," 2003.

[7] N. M. Scala and J. A. Pazour, "A Value Model for Asset Tracking Technology to Support Naval Sea-Based Resupply," *Engineering Management Journal,* vol. 28, no. 2, pp. 120-130, 2016.

[8] P. J. Reilly, J. A. Pazour and K. R. Schneider, "Propagation of unit location uncertainty in dense storage environments," *International Journal of Production Research,* vol. 55, no. 18, pp. 5435-5449, 2017.

[9] M. Awwad and J. A. Pazour, "Search plan for a single item in an inverted T k-deep storage system," *Military Operations Research,* vol. 23, pp. 1-8, 2018.

[10] M. Awwad and J. A. Pazour, "Batch order picking in a dense storage system with item location uncertainty," in *Proceedings from Industrial and Systems Engineering Research Conference*, Pittsburgh, 2017.

[11] J. A. Pazour and H. J. Carlo, "Warehouse reshuffling: Insights and optimization," *Transportation Research Part E: Logistics and Transportation Review,* vol. 73, pp. 207-226, 2015.

[12] H. J. Carlo and G. E. Giraldo, "Toward perpetually organized unit-load warehouses," *Computers & Industrial Engineering,* vol. 63, no. 4, pp. 1003-1012, 2012.

[13] K. R. Gue and B. S. Kim, "Puzzle based storage systems," *Naval Research Logistics,* vol. 54, no. 5, pp. 556-567, 2007.

[14] N. Zaerpour, Y. Yu and R. de Koster, "Small is beautiful: A framework for evaluating and optimizing live-cube compact storage systems," *Transportation Science,* vol. 51, no. 1, pp. 34-51, 2015.

[15] K. R. Gue, K. Furmans, Z. Seibold and O. Uludag, "GridStore: a puzzle based storage system with decentralized control," *IEEE Transactions on Automation Science and Engineering,* vol. 11, no. 2, pp. 429-438, 2014.

[16] A. F. Archer, "A modern treatment of the 15 puzzle," *The American Mathematical Monthly,* vol. 106, no. 9, pp. 793-799, 1999.

[17] T. Lamballais and M. De Koster, "Estimating performance in a robotic mobile fulfillment system," *European Journal of Operational Research,* vol. 253, no. 3, pp. 976-990, 2017.

[18] N. Boysen, D. Briskorn and Emde, Simon, "Parts-to-picker based order processing in a rack-moving mobile robots environment," *European Journal of Operational Research,* vol. 262, no. 2, pp. 550-562, 2017.

[19] Y. A. Bozer and F. J. Aldarondo, "A simulation-based comparison of two goods-to-person order picking systems in an online retail setting," *International Journal of Production Research,* pp. 1-21, 2018.

[20] H. J. Carlo, I. F. Vis and K. J. Roodbergen, "Storage yard operations in container terminals: Literature overview, trends, and research directions," *European Journal of Operational Research,* vol. 235, no. 2, pp. 412-430, 2014.

[21] K. H. Kim and H. Hwand, "Sequencing delivery and receiving operations for yard cranes in port container terminals," *International Journal of Production Economics,* vol. 84, pp. 283-292, 2003.

[22] I. F. Vis, "A comparative analysis of storage and retrieval equipment at a container terminal," *International Journal of Production Economics,* vol. 103, no. 2, pp. 680-693, 2006.

[23] A. H. Gharehgozli, F. G. Vernooij and N. Zaerpour, "A simulations of the performance of twin automated stacking cranes at a seaport containter terminal," *European Journal of Operational Research,* vol. 261, no. 1, pp. 108-128, 2017.

[24] P. S. Bender, "Mathematical modeling of the 20/80 rule: theory and practice," *Journal of Business Logistics,* vol. 2, pp. 139-157, 1981.

Chapter 6: Search plan for a single item in an inverted T k-deep storage system

This chapter has been published in the following peer-reviewed journal:

1. Awwad, Mohamed*, and Pazour, Jennifer A., 2018, "Search plan for a single item in an inverted T k-deep storage system," *Military Operations Research*, 23, 1-18.

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

We study the problem of searching for a single item in a dense storage system 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. An optimization model and heuristic solution approach to minimize the expected search effort uses developed approximations to consider how to conduct repositioning and put-back operations, in addition to traveling, within a search procedure. Given an inverted T k-deep storage system, 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. Our analysis shows that the search process in dense storage systems have high variability and thus using the full distribution of search times is recommended for downstream planning.

INTRODUCTION

The two primary objectives for 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.

Existing warehousing literature (Gu, Goetschalckx, and McGinnis (2007)) 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. This 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. Combining searching with the traditional order picking routing problem is novel and to our knowledge has not been studied before. Searching in a dense storage environment is particularly challenging because it can require moving items out of the way during the search process. Thus, there are three elements associated with the search procedure in a dense storage environment: the travel component, the repositioning effort carried out by the searcher to gain access to deeply stored items, and the put-back effort to move back repositioned items to their old storage locations before continuing with 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. Sea bases act as floating distribution centers "providing ready issue material to forces ashore" participating in a variety of missions (Clark, 2002). Sea bases have limited space and require the use of dense storage in the ships' storerooms. In addition, 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). 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 to investigate the possibilities of conducting safe and effective joint operations on or from the sea (CJOSCOE, 2012). The US Navy has made progress in understanding and designing the sea base concept. However, the best way to conduct distribution and supply operations to "resupply combat troops from the sea" is still an open challenge (CJOSCOE, 2012; Parsons, 2013).

Due to the unique characteristics of sea base operations, imperfect visibility of items' locations exists. Specifically, the requirements of both dense storage and selective offloading capabilities, result in item location uncertainty not seen in traditional aisle based systems (Pazour & Shin, 2016; Reilly, Pazour, & Schneider, 2017). Also, asset tracking technologies, like radio frequency identification (RFID), provide only binary asset tracking information and do not provide coordinate information of items. Given the need for a large number of items stored densely, the use of license plates -- such as the ones used in a traditional aisle-based storage system -- are also not applicable. Consequently, existing asset tracking technologies, like RFID, do not identify item locations to the granularity required to eliminate imperfect visibility when dense storage and selective offloading are required (Scala & Pazour, 2016). In addition, existing studies have found even with an aisle-based storage system with RFID tracking, searching can be required because item location uncertainty exists (Hariharan & Bukkapatnam, 2009). Asset visibility onboard of sea base ships is a strategic issue still being investigated by the Department of Defense (DoD). Its main objective is to "enhance the asset visibility in a manner that provides the ability to track assets throughout their lifecycle" (Peters, 2014).

The purpose of this research is to study search procedures in dense storage environments. Our contributions are three-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 during searching to gain access to other deeply stored items. Therefore, the search process needs to consider 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 or other deeply stored items
2. Formulate the warehouse search problem as a mixed integer linear program. Our contribution is modeling focused, as we model the problem of searching in dense storage and show how the problem can be broken down into the time elements that can be approximated a priori and used as input parameters to the optimization formulation. In addition, we provide a heuristic solution approach for the optimization formulation.
3. Provide insights about the characteristics of searching in very high density storage systems with item location uncertainty. Two repositioning policies are considered. The first 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 or access to another deeply stored item. The second policy assumes items are repositioned outside the storeroom grid. This 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.

The remainder of this paper is organized as follows: the next section surveys the literature on dense storage and search theory. Next we describe the search environment (e.g., the inverted T dense storage layout), provide an optimization formulation, denote the assumptions of the model, and provide closed-form expressions for the search components. Then we provide an item search heuristic and present a summary of the numerical experiments and statistical analysis. Finally, we provide conclusions and suggestions for extending this research.

LITERATURE REVIEW

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

Dense Storage

Engineers working in warehousing and distribution industries have developed ways to increase the storage density (Gue, 2006). Several publications, such as Rana (1990), increase warehouse storage density through dedicating less aisle space by using narrow or very narrow aisles. However, Parikh and Meller (2010) identified that such increases in density must be traded-off with travel congestion and restricted travel in one-way aisles. A second way to increase storage density is to use 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 of one item is required to access the other. One form of the k-deep systems is the double-deep automated storage and retrieval system discussed by Lerher, Sraml, Potrc, and Tollazzi (2010). A third way 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 block stacking models, 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) defines a storage system as very high density 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 to retrieve the one behind it. Contrastingly, a single-deep storage system is not considered a very high density system, as all pallets are accessible directly from aisles. The letter k will refer in this work to the accessibility constant. When the very high density storage system is automated, it is usually called a deep bulk storage system (Gue, 2006). In this work, we study manual very high density storage environments with inverted T configurations, which have one horizontal and one vertical aisle (Gue, 2006). The highest storage density for a fixed value of the accessibility constant k is achieved in layouts that resemble the inverted T configuration (Futcher, 2003).

Search Theory

Search theory originated from the work of Koopman (1956) and others during World War II. It determines 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. Stone (1983), studying different elements of search planning, presents an approach that encompasses both objective and subjective methods. A compromise between theoretically optimal plans and operationally feasible ones is provided.

The problem we study here can be classified as a stationary discrete space target search problem. It is the problem of finding a stationary target located 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). Other search problems fall under search planning in a continuous space and time with a moving target such as the work of Kierstead and DelBalzo (2003). One of the main differences between existing search theory problems and the studied problem is that while searching in a dense storage environment items have to be moved out of the way in order to gain access to another one.

Searching in a traditional aisle-based warehousing environment is the focus of Hariharan and Bukkapatnam (2009). They develop a search model based on a 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. They consider a traditional aisle based storage system, whereas we study a dense storage system with an inverted T configuration that requires considering the movements to reposition items to gain access to other items while searching.

INVERSE-T DENSE STORAGE ENVIRONMENT AND COORDINATE SYSTEM

In this work we consider an inverted T dense storage system that is operated manually. The exact locations of stored items are assumed 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.

Before presenting an optimization model, the inverted T dense storage layout and a coordinate system is defined. An inverted T configuration is a configuration with one horizontal aisle and one vertical aisle as displayed in Figure 1. This configuration is recommended by Gue (2006) because it can 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. In our work, all items are assumed one unit high in the vertical space direction, i.e. stacked storeroom configurations are not considered.

To calculate the time required for searching, a coordinate system is defined. The origin point ($x = 0$ and $y = 0$) is located at the intersection between the vertical and horizontal aisles. The input/output (IO) 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 it reaches $-k$, and increases in the positive direction until the total number of rows is equal to h, where h is the storeroom length. To maintain the inverted T configuration, h has to be greater than or equal to $2k + 2$. Using this coordinate system, the two variables k and h are the only two parameters needed to define the storeroom configuration. Figure 1 illustrates a 5 by 6 2-deep inverted T storeroom with coordinates defined for each cell; consequently, $k = 2$ and $h = 6$.

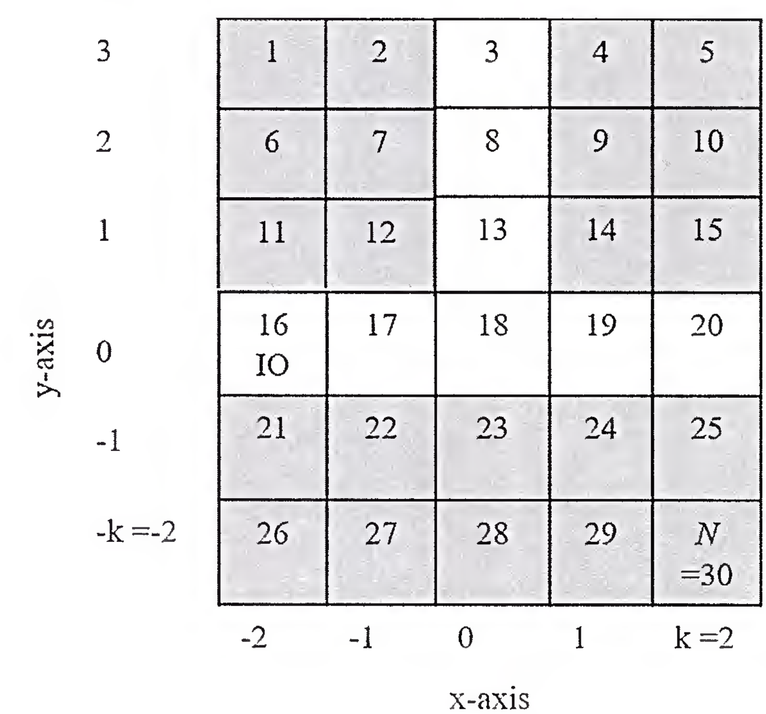


Figure 1: Sample inverted T system with $\mathbf{k} = \mathbf{2}$ and $\mathbf{h} = 6$; shaded cells (occupied), unshaded (unoccupied).

The search region for our problem is a discrete environment of non-overlapping cells of equal sizes (i.e., one cell can hold one pallet). 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. Thus, the storeroom inverted T configuration remains the same after each retrieval of an item.

Any storeroom is divided into a number of N cells and stores a number n pallets in it, where $n \leq N$. We define the N cells as follows. The set C is the set of cells indexed on $c = 1,2, \dots, N$. The set of cells C consists of the two disjoint subsets: the set of occupied cells C_o indexed on $c_o = 1,2, \dots, n$ 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 $y = -k$.

Each configuration's maximum storage density can be calculated as $\frac{n}{N} = \frac{h(2k+1)-(2k+1)-(h-1-k)}{h(2k+1)}$. In Figure 1, the number of cells is $N = 30$, the maximum number of occupied cells with pallets is $n = 22$, and the associated storage density is 73.3%. Reconfiguring an entire storage area of a naval ship into an inverted T configuration may not always be possible due to the shape of the ship and the presence of obstacles. Therefore, the storage area can be configured into multiple smaller inverted T configurations beside each other, or some of the occupied cells can be left open to accommodate the irregularities in the ships' storage areas.

PROBLEM FORMULATION

The problem under study is a search planning problem with the objective of minimizing the expected 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 based on the summation of the time needed to travel between a pair of unoccupied 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 that impede the IO point to their storage location before traveling to the next cell (if any). The decision variables in our problem are the search sequence of occupied cells. In this work, we assume put-back can either be approximated by a put-back rule based on repositioning or is assumed to occur after finding the requested item. While we capture the time to put-back pallets in our analysis, when to conduct put-back is not considered a decision variable. An input to the optimization problem is the search time weight between occupied cells $i \in c_o$ with coordinates x_i and y_i and occupied cell $j \in c_o$ with coordinates x_j and y_j and is denoted as $T_{(x_i,y_i),(x_j,y_j)}$ and calculated as follows:

$$T_{(x_i,y_i),(x_j,y_j)} = \tau_{(x_u,y_u),(x_v,y_v)} + R_{(x_j,y_j)} \quad (34)$$

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. The locations u and v are functions of the locations i and j, respectively. $R_{(x_j,y_j)}$ denotes the repositioning time to gain access to the pallet located at cell j and assumes that pallets in front of other pallets have already been repositioned. In subsequent sections, we provide analytical models to either calculate or approximate these time components, such that T_{ij} can be considered independent of the search sequence and used as an input parameter to our optimization model and search heuristic.

The following assumptions are made about the search process. The search process occurs in the inverted T dense storage layout described in Section 3. Thus, orders are received at the given IO location to find and retrieve a single item located in one of the n occupied cells. All pallets are present in their original locations before being repositioned to either open aisle locations or outside the storeroom. No items are located initially in an unoccupied cell. When the search process starts, the probability the searcher will find the item requested is given and known for each occupied cell. 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. A single searcher is considered and the searcher can guarantee a target item's presence or absence after the cell has been searched for a negligible amount of time and is 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.

It is assumed that pallets located at any occupied cell $j \in C_O^{UL} \cup C_O^{UR}$ are accessed only through the vertical aisle and if repositioned inside the storeroom, 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 j $\in C_O^{BL} \cup C_O^{BR} \cup C_O^{BC}$) are accessed only through the horizontal aisle and are always repositioned to the open locations in the vertical aisle (if repositioned inside the storeroom). In addition, we assume put-back occurs after each set of deeply stored items are repositioned. For example, for pallets in the upper left or right part of the storeroom, put-back would occur after the k-1 pallets (all from the same y-coordinate) are repositioned. This allows us to estimate put-back using repositioning calculations and prevents extra repositioning effort that would occur if put-back occurred after every move.

Optimization Formulation

In this section we present a mixed integer linear programming (MILP) model for the problem of searching for a single item in a dense storage environment with an inverted T configuration. The problem of searching for a single item in a dense storage system resembles the problem of scheduling n jobs to be processed on a single machine or a Job Shop Scheduling Problem with sequence-dependent processing times (Bianco et al, 1988). Other problems formulated in a manner resembling the JSSP include the formulation of Ascheuer, Fischetti, and Grötschel (2001) of the Asymmetric Travelling Salesman Problem with time windows. The formulation relies on the set of occupied cells, c_o indexed on i, j. We let $c_o^{IO} = c_o \cup IO$, which is the union of the set of occupied cells with the IO cell. This formulation requires a "dummy" cell (which we denote as *D*), because the last cell does not precede any other cell in the search sequence. We denote the union of the set of occupied cells, the IO cell, and the dummy cell as $c_o \cup IO \cup D$.

The input parameters for the optimization formulation include T_{ij} which is the search (travel plus repositioning) time to go from occupied cell i with coordinates x_i and y_i to occupied cell j with coordinates x_j and y_j (and calculated in (1)). Therefore, the optimization formulation captures travel and repositioning time, and assumes that put-back either occurs outside of the search routine (e.g., after an item is found similar to repositioning policy 2) or is approximated as part of its repositioning time. When to conduct put-back is not considered as a decision variable in this formulation. For pallets with the same nearest accessible open aisle location, more deeply stored pallets will always have larger travel and repositioning times than less deeply stored pallets. Therefore, no constraints are needed to enforce that repositioning calculations assume that pallets in front of other more deeply stored pallets have already been repositioned. Note, $T_{IO,j}$ is the search time to go from the IO point to occupied cell j. Also, the input parameters associated with the dummy cell D are $T_{i,D} = 0 \; \forall i \in c_o^{IO}; and \; T_{D,i} = M \; j \in c_o^{IO}$. Here, *M* denotes a very big number. An additional input parameter is π_i which denotes the probability we find the item we are searching for in occupied cell *i*.
The decision variables are defined as,

- Z_{ij} is 1 if cell $i \in c_o^{IO}$is directly preceded in the search sequence by occupied cell *j*; 0 otherwise
- S_i is the search time of an occupied cell $i \in c_o^{IO}$, which given the cumulative nature of search, also denotes the time that the searcher arrives to cell $i \in c_o^{IO}$ Note, S_{IO} is the start time at the IO point and is an input parameter set equal to 0.
- Additional record keeping variables, Y_{ij} ,are used to decouple the Z_{ij} and S_i variables, allowing for a linear formulation. Specifically, Y_{ij} is 1 if $Z_{ij}=0$; and is 0 if $Z_{ij}=1$ or $Z_{ij}=0$.

Equations (2) – (10) provide an optimization model that minimizes the expected search time.

$$minimize \sum_{i\in c_o^{I/O}} S_i\pi_i$$
(2)

$$S_i + \sum_{j\in c_o^{IO} \cup D} T_{ij}Z_{ij} - \sum_{k\in c_o^{I/O}} \sum_{j\in c_o^{IO} \cup D} T_{kj}Z_{kj} \leq 0 \; \forall \, i \in c_o^{IO}$$
(3)

$$-(S_j - S_i - T_{ij}) \leq MY_{ij} \; \forall \, i \in c_o^{IO} \; \forall j \in c_o^{IO} \cup D : j \neq i$$
(4)

$$Z_{ij} \leq M(1 - Y_{ij}) \; \forall \, i \in C_o \cup 0 \; \forall \, i \in c_o^{IO} \; \forall j \in c_o^{IO} \cup D : j \neq i$$
(5)

$$\sum_{j\in c_o^{I/O} \cup D} Z_{ij} = 1 \; \forall i \in c_o^{IO} \qquad (6)$$

$$\sum_{i\in c_o^{I/O}} Z_{ij} = 1 \; \forall j \in c_o^{IO} \cup D \qquad (7)$$

$$Z_{ij} = \{0,1\} \; \forall i \in c_o^{IO} \forall j \in c_o^{IO} \cup D \qquad (8)$$

$$Y_{ij} = \{0,1\} \; \forall i \in c_o^{IO} \forall j \in c_o^{IO} \qquad (9)$$

$$S_i \geq 0 \; \forall i \in c_o^{IO} \cup D \qquad (10)$$

The objective, Equation (2), is to minimize the expected time to search over all occupied cells. Equation (3) ensures that the arrival time to each cell must be the sum of the arrival time of previous cells plus the search time of the previous searched cell. Equation (5) ensures that when $Z_{ij} = 1$, then $Y_{ij}=0$, and in equation (4) when $Y_{ij}=0$, it enforces that a searcher cannot start searching a cell until the previous cell search time. When $Z_{ij} = 0$, then no restrictions are placed on Y_{ij} and equation (4) is redundant. Equations (6) and (7) enforce that at most one occupied cell can be in each sequence location. The remaining equations enforce variables bound restrictions.

TIME COMPONENT EXPRESSIONS

In the next subsections, we discuss how we are able to break up the problem and provide analytical expressions to calculate the input parameters T_{ij} for the search times between cells i and j.

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 and $|y_i| > k$, 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 and $|y_i| < k$, 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 1 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 $|y_4| > k$. 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. Cell 4 belongs to the set of upper occupied cell C_O^U with $|y_4| > k$, therefore, the nearest open aisle location is located at $x_u = 0$ and $y_u = y_4 = 3$, i.e. cell 3. Finally, calculate the travel time distance between cells 18 and 3 as:

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

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. Repositioning is required only for cells that do not have direct contact with an open aisle. The remaining cells j have direct contact with open aisle locations and do not require repositioning; $R_j = 0$ for all occupied cells with $j \in C_O^B$ and $y_j = -1$; or $j \in C_O^U$ and $|x_j| = 1$.

For the remaining occupied cells, we investigate two repositioning policies. 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

IO 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 one at a time inside the storeroom to unoccupied open cells. We follow the repositioning rule proposed by Futcher (2003), which can be summarized as follows:

- Reposition a pallet that blocks the target pallet's j way to the IO location in an open aisle location that does not impede the target pallet's path to the IO point.

This repositioning rule moves a pallet to the closest empty aisle location that does not impede the path from the pallet we need to gain access to and the IO point. 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}.

As a reminder, $R_{(x_j,y_j)}$ denotes the repositioning time to gain access to the pallet located at cell j. The repositioning equations are based on the cell the searcher needs to access and not the cell that is repositioned. Also, repositioning equations assume that pallets in front of other pallets have already been repositioned (e.g., the repositioning calculation for a pallet located in an upper right cell with $x = 3, y = 2$ coordinates assumes that pallet $x = 2, y = 2$ has already been repositioned). To develop repositioning distances equations that can be calculated a priori, the repositioning distances represent a worst-case expression based on temporary repositioning locations being the farthest open aisle location based on their depth. For example, repositioning items in the upper occupied cell locations that have x-coordinates of +/-1 are assumed to be repositioned to the unoccupied cell in the horizontal aisle with x-coordinate of k. This allows the opportunity for additional items to be repositioned in the aisle, but sometimes overestimates the repositioning.

For example, in Figure 1 the item in cell 2 is repositioned to gain access to the item in cell 1. The item in cell 2 is temporarily repositioned to cell 20. The put-back policy occurs after $k - 1$ items have been repositioning. For Figure 1, with $k = 2$, this would occur after item 1 was searched. For a configuration with $k = 3$, put-back would occur after all deeply stored items with the same y-coordinate (for occupied upper cells) have been searched. This put-back policy is chosen in order to avoid having a repositioned item impeding the searcher's way to an unvisited search location. For example in Figure 1 if the item in cell 2 was temporarily located in cell 20, and the next search location is cell 25, the searcher will not be able to visit cell 25 unless the item blocking the way was put-back to its storage location.

The following equations calculate the repositioning distance to gain access to the item in cell j. As a reminder, these repositioning distance equations are based on the assumption that the searcher's access to deeply stored items stored above the horizontal aisle of the storeroom can only be accessed through the vertical aisle, and access to deeply stored items stored below the horizontal aisle are accessed through the horizontal aisle. Also, the equations are based on repositioning only the item in the cell immediately blocking access to cell *j*.

For any item located at cell j $\{x_j, y_j\}$ that belongs to the set C_O^{UL} of occupied cells in the upper left part of the storeroom with $x_j \leq -2$; or belongs to the set C_O^{UR} of occupied cells in the upper right part of the storeroom with $x_j \geq 2$; or belongs to the set C_O^{BR} of occupied cells in the bottom right part of the storeroom with $y_j \leq -2$; then the total repositioning time required to gain access to item *j* is equal to

$$R_{(x_j,y_j)} = 2 \times [1 + |y_j| + |x_j|] \tag{11}$$

Where, 2 is the multiplicative factor accounting for the searcher's trip from repositioned item *j* repositioning location to its original storage location. For example, with a *k=3* configuration, to gain access to cell (2,3), equation (11) calculates the distance to move the item in cell (1,3) to cell (2,0), which is 12 units. To gain access to the item in cell (3,3), equation (11) calculates the distance to move the item in cell (2,3) to worst-case assumption cell (2,0), which is 14 units.

For any item 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 with $y_j \leq -2$, the total repositioning time required to gain access to item *j* is equal to

$$R_{(x_j,y_j)} = 2 \times [|y_j| + |x_j|] \tag{12}$$

For any item 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 with $y_j \leq -2$, the total repositioning time required to gain access to item *j* is equal to

$$R_{(x_j,y_j)} = 2 \times |y_j| \tag{13}$$

<u>Policy 2: Repositioning occurs outside the storeroom</u>

In this policy, items that need repositioning are repositioned outside the storeroom grid to a fixed location assumed to have coordinates $x = -k - 1$ and $y = 0$ as shown in Figure 2. This repositioning policy represents an environment where the repositioned items are left outside the storeroom until the item we are looking for is found in the main storeroom and retrieved to the IO location.

The main motivation of repositioning items outside the storeroom grid is to avoid the need to put-back repositioned items to their old storage locations while the search procedure is being carried out. Instead, putting back of repositioned items 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 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 assumes there is available space outside the storeroom to temporarily store repositioned items to the fixed location. Also, it assumes available underutilized labor to put the items back after the search process is completed.

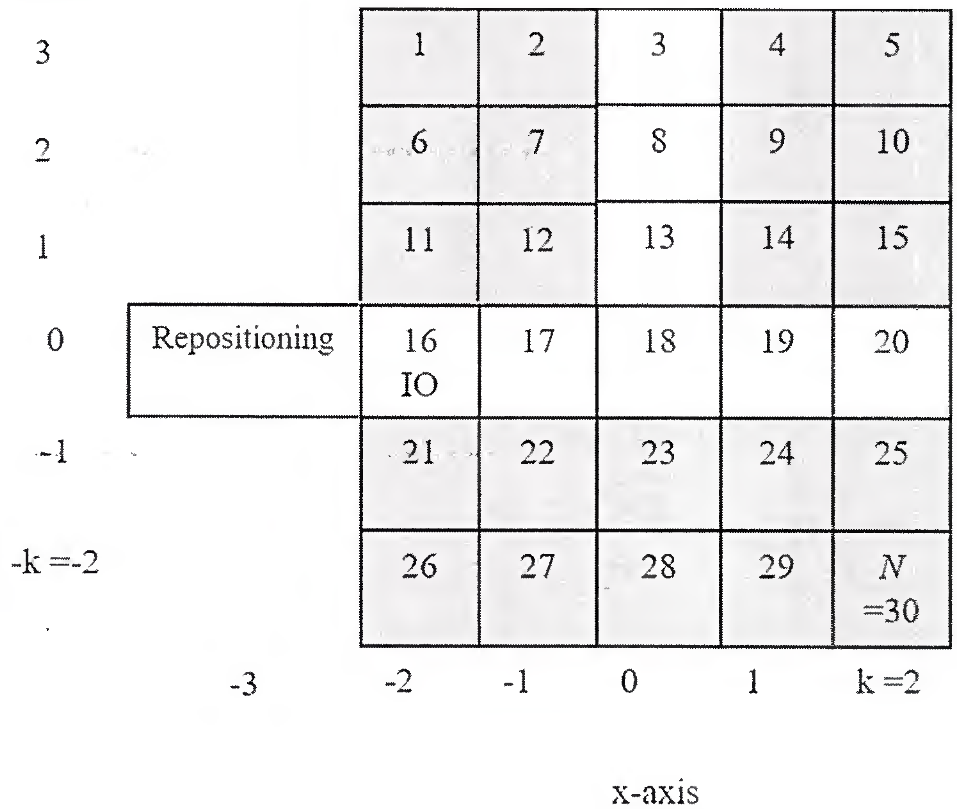


Figure 2: Repositioning outside the storeroom grid with *k=2* and *h=6.*

As items are repositioned to a certain fixed location, the repositioning distance equations are different from those in repositioning policy 1. For any item located at cell j $\{x_j, y_j\}$ belonging to the sets $C_O^{BR}, C_O^{BC}, C_O^{BL}$ with $y_j \leq -2$ or a cell j belonging to C_O^{UR} with $x_j \geq 2$ the repositioning distance is given by

$$R_{(x_j,y_j)} = 2 \times \left[|y_j| + |x_j + k + 1| - 1\right] \quad (14)$$

For any item located at cell j $\{x_j, y_j\}$ belonging to the set C_O^{UL} with $x_j \leq -2$, the repositioning distance is given by

$$R_{(x_j,y_j)} = 2 \times \left[y_j + |x_j| + k + 1\right] \quad (15)$$

Put-back Time

The purpose of a put-back policy is to move items that have been temporally repositioned to an unoccupied cell back to their occupied location so that neither the searcher's path to the IO point nor access to an unvisited location is impeded. This will occur only in the first repositioning case. In the second repositioning case, the items repositioned outside the storeroom do not impede the searcher's path, and 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:

- Put-back occurs after each set of deeply stored items are repositioned. For example, for pallets in the upper left or right part of the storeroom, put-back would occur after the k-1 pallets (all from the same y-coordinate) are repositioned. For pallets in the bottom part of the storeroom, put-back would occur after the k-1 pallets (all from the same x-coordinate) are repositioned.

The above rule is established because it allows us to estimate put-back using repositioning calculations and prevents extra repositioning effort that would occur if put-back occurred after every move. Also, any further repositioning before putting back the existing repositioned item might impede the path of the searcher to unvisited cells in the search path or an item being retrieved to the IO location. Priority is given to retrieve the target item if found before put-back of the repositioned item to its storage location. The decision whether to initiate a put-back or not is made when the number of repositioned items is equal to k-1, and before traveling to the next cell in the search procedure. Thus, put-back time is a function of the search sequence. In order to consider put back independent of search sequence, put-back time of cell j (P_{bj}) is approximated as being equal to its repositioning time of cell j. Therefore, put-back distances are calculated using the same set of equations under *case 1* used to calculate the repositioning distances.

SEARCH PLAN HEURISTIC

The search plan optimization problem can be reduced to a Job Shop Scheduling Problem with sequence-dependent processing times, which is known to be NP-Hard (Bianco et al, 1988). Thus, the search plan optimization problem is NP-Hard and we develop a search plan heuristic. The search plan heuristic is based on a network representation that uses a greedy approach to determine the sequence of occupied cells to visit. It starts by generating arcs from the current cell (initially, when a new order is received, the current cell is the IO cell) and evaluating the weights of each of those arcs, which are a function of travel and repositioning times, and location probability. The indices of the cell connected with the arc with the least weight are added to the search sequence array Ψ of visited cells. This procedure is repeated until the target item has been detected.

Given items that have direct contact with open aisle locations do not require repositioning, a repositioning subroutine is defined to track whether the w^{th} item searched requires repositioning through a binary repositioning items array *Repo*. The search heuristic does not consider put-back when selecting arcs and instead evokes the subroutine *Put-back* after *k-1* items have been repositioned. The subroutine *Put-back* counts the number of repositioned items still in their temporary locations using the *Repo* array. Having the sequence Ψ with the indices of visited cells and their coordinates, the expected time taken to find a specific item in the storeroom, which is the sum of travel, repositioning, and put-back time, can be calculated by incorporating the probability of finding an item in cell *i* (given as π_i). In addition, Ψ can be mapped to decision variables Z_{ij} given in the optimization formulation. The search plan heuristic for case 1, where pallets are repositioned inside the storeroom is summarized through the following pseudo code:

Algorithm: *search*

1. Given: k and h, and *ItemNumber* of item searching for.
2. Generate x and y coordinates for each cell $i \in C$, and classify each cell $i \in C$ in its appropriate subset ($C_O^{UL}, C_O^{UR}, C_O^{BL}, C_O^{BR}, C_O^{BC}, C_U$). Set the probability of finding the item for each occupied cells as π_i
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, an empty set of visited cells array ψ, an empty Repo array, and empty S array. Set $w = 0, \alpha = 1, \beta = 0$ and $Time = 0$.
5. Start the search procedure at IO cell, i.e. $\Psi[w] = IO$ cell location.
6. Set $i = IO\ cell.\ \psi = \psi + IO; w = w + 1.$
7. Calculate the distance $T_{(x_i,y_i),(x_j,y_j)} = \tau_{(x_u,y_u),(x_v,y_v)} + R_{(x_j,y_j)}$ using equation (1) between cell i and cell $j\ \forall\ j \notin \psi$ using travel and repositioning equations for *case 1* as given in Section 5.
8. Select cell $j^*$with min $\pi_j\ T_{ij}$; break ties arbitrarily.
9. Update $\Psi[w] = j^*, \psi = \psi + j^*$, and $S[w] = Time + T_{ij*}$
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. If ($ItemNumber = \Psi[w]$)

    Then stop *search*

    Output: Ψ and $Time$.

    Else

    Do Subroutine *Put-back*

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

Subroutine Reposition

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

Then start repositioning,

set $Repo[w] = 1$, and calculate R_{j^*} by equation (11).

Search more deeply stored items:

For *l=1* to *k-2:*

w=w+1.

Select cell m with coordinates $(x_{j^*} - l, y_{j^*})$.

Set $Repo[w] = 1$; calculate R_m by equation (11).

Update $\Psi[w] = m$,$\psi = \psi + m$, and $S[w] = Time + T_{j*,m}$.

If ($ItemNumber = \Psi[w]$)

Then stop *search*

Output: Ψ and $Time$.

Loop

Set j*=m

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

Then start repositioning,

set $Repo[w] = 1$, and calculate $R_{j^{**}}$ by equation (11).

Search more deeply stored items:

For *l=1* to *k-2:*

w=w+1.

Select cell m with coordinates $(x_{j^*} + l, y_{j^*})$.

Set $Repo[w] = 1$; calculate R_m by equation (11).

Update $\Psi[w] = m$,$\psi = \psi + m$, and $S[w] = Time + T_{j*,m}$.

If ($ItemNumber = \Psi[w]$)

Then stop *search*

Output: Ψ and $Time$.

Loop

Set j*=m

If ($cell\, j \in C_o^{BR}$) AND ($y_{j^*} \leq -2$)

Then start repositioning,

set $Repo[w] = 1$, and calculate $R_{j^{**}}$ by equation (11).

Search more deeply stored items:

For *l=1* to *k-2*

w=w+1;

Select cell m with coordinates $(x_{j^*}, y_{j^*} - l)$.

Set $Repo[w] = 1$; calculate R_m by equation (11).
Update $\Psi[w] = m, \psi = \psi + m$, and $S[w] = Time + T_{j*,m}$

If $(ItemNumber = \Psi[w])$
Then stop *search*

Output: Ψ and $Time$

Loop
Set j*=*m*

If (cell $j^* \in C_O^{BL}$) AND $(y_{j^*} \leq -2)$
Then start repositioning,
set $Repo[w] = 1$, and calculate $R_{j^{**}}$ by equation (12).
Search more deeply stored items:
For *l=1* to *k-2*
w=w+1
Select cell m with coordinates $(x_{j^*}, y_{j^*} - l)$.
Set $Repo[w] = 1$; calculate R_m by equation (12).
Update $\Psi[w] = m, \psi = \psi + m$, and $S[w] = Time + T_{j*,m}$

If $(ItemNumber = \Psi[w])$
Then stop *search*

Output: Ψ and $Time$

Loop
Set j*=*m*

If (cell $j^* \in C_O^{BC}$) AND $(y_{j^*} \leq -2)$
Then start repositioning,
set $Repo[w] = 1$, and calculate $R_{j^{**}}$ by equation (13).
Search more deeply stored items:
For *l=1* to *k-2*
w=w+1.
Select cell m with coordinates $(x_{j^*}, y_{j^*} - l)$.
Set $Repo[w] = 1$; calculate R_m by equation (13).
Update $\Psi[w] = m, \psi = \psi + m$, and $S[w] = Time + T_{j*,m}$

If $(ItemNumber = \Psi[w])$

Then stop *search*

Output: Ψ and $Time$.

Else

Loop

Set j*=m

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

Subroutine Put-back

For $\sum_{l=0}^{w} Repo[l] = \alpha\,(k-1)$

Then do put-back for last k-1 repositioned items, approximating put-back time of cell j (P_{bj}) using the appropriate repositioning equation.

Set $\alpha = \alpha + 1$, and $Time = Time + P_{bj}$.

Else no Put-Back required

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

OPERATIONAL SEARCH PLAN

Applying the search heuristic from the previous section will generate the search array Ψ, which provides the sequence of cells to search. Because the target is stationary and the probability the target item is in any occupied cell is stationary, the sequence of the cells in the search sequence does not change for a given problem instance. In certain environments, the search array Ψ can be used to provide an operational search plan that can be articulated to a human searcher. For reasonably shaped storerooms with an equal chance of finding an item in one of the occupied cells, we recommend the searcher to search all cells that can be accessed through open aisle locations before searching cells that require repositioning items to temporary open aisle locations to gain access to a deeply-stored pallet. All locations that face an aisle do not require repositioning or put-back time and thus for practical storeroom instances (e.g., when the length h is not extremely larger than k), aisle-facing locations are more accessible than items stored more deeply.

Although the specific search sequence Ψ will differ for the two repositioning cases (because the repositioning distances/equations are different), the recommended operational policy still applies. The main difference between the two search sequences appears after visiting all cells in contact with open aisle locations. In the first repositioning case, the searcher will visit deeply-stored cells in an organ pipe structure, starting from the bottom center and working out. As an example, Figure 3 shows the search sequence of a storeroom with a configuration of $k = 3$ and $h = 7$, where each cell is numbered in the order it is visited.

y-axis \ x-axis	-3	-2	-1	0	1	2	k=3
3	39	38	16		17	36	37
2	35	34	14		15	32	33
1	1	3	5		8	10	12
0	IO						
-1	2	4	6	7	9	11	13
-2	28	24	22	18	20	26	30
-k=-3	29	25	23	19	21	27	31

Figure 3: The search sequence according to repositioning policy 1 in storeroom with $\boldsymbol{k} = 3$ *and* $\boldsymbol{h} = 7$*, (numbers in cells are the order to search).*

In the second case, the searcher will visit deeply-stored cells with low values of repositioning distance relative to the fixed repositioning location with coordinates $x = -k - 1$ and $y = 0$. For example, Figure 4 exhibits the search sequence, for the second repositioning strategy, applied to a storeroom with the same configuration as in Figure 3. After cells on the aisle locations are visited, next deeply stored cells closest to the IO point are visited.

y-axis								
3		39	38	16		17	36	37
2		35	34	14		15	32	33
1		1	3	5		8	10	12
0	Repositioning	IO						
-1		2	4	6	7	9	11	13

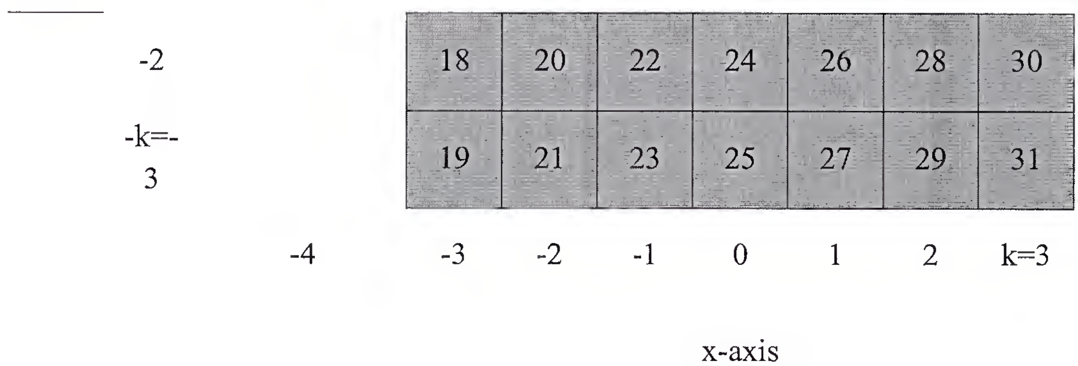


Figure 4: Search sequence according to repositioning policy 2 in storeroom with $\boldsymbol{k} = \mathbf{3}$ *and* $\boldsymbol{h} = \mathbf{7}$.

While these operational search plans can be provided to human searchers, they are not applicable if the probability of finding an item is not uniform. For example, if the chance to find an item is high enough, even if it is deeply stored, the heuristic will recommend to visit that location before all aisle locations have been visited. Also, if the storeroom is very tall (i.e., h >> k), there may be aisle locations that are part of the upper-left or upper-right part of the store room that will be recommended to be visited after deeply stored locations in the bottom part of the store room.

NUMERICAL EXPERIMENTS

In this section, we conduct numerical experiments to better understand the characteristics of searching in dense storage systems. For each of the generated problem instances we use the search plan heuristics to obtain a search sequence that is then used to calculate: (1) the search time (sum of travel, repositioning, put-back) to find an item in any given occupied cell $i \in c_o$, (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. 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. We assume that each occupied cell is equally likely to contain the requested item, thus, $\pi_i = 1/n$. The search heuristic 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 both repositioning policies, Figure 5 exhibits the search time for each *n=38* occupied cells in a storeroom configurations with $k = 2$ and $h = 10$. The y-axis is the search time for each cell. The occupied cells are presented from left to right in increasing order based on search time. Thus, the left-most data point is the first cell in the search sequence, and the last data point is *n=38* in the search sequence, since there are 38 occupied cells in the storeroom configuration

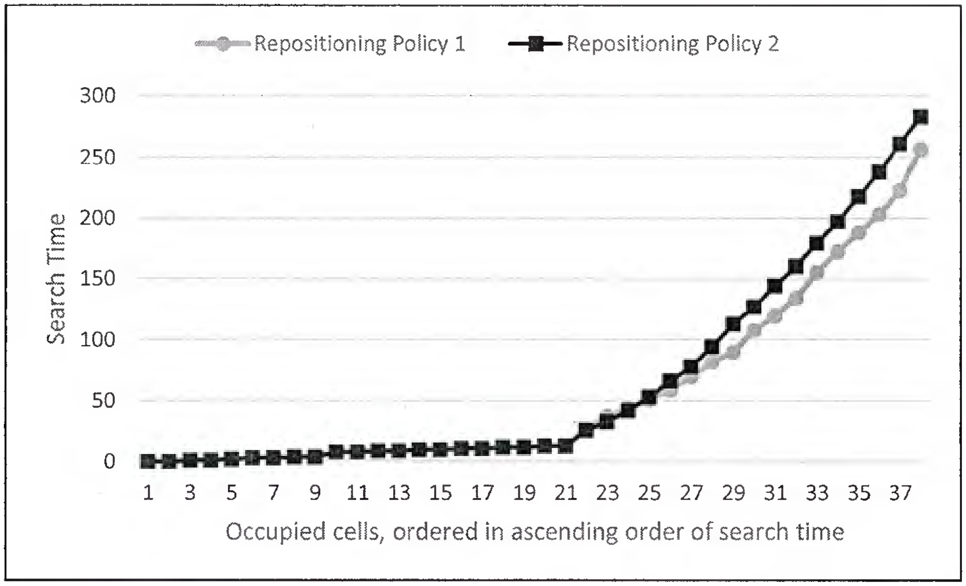


Figure 5: Search Time versus the Number of Pallets for a storeroom with $k = 2$ *and* $h = 10$ *for repositioning policy 1 in yellow and repositioning policy 2 in blue.*

Both curves in Figure 5 coincide with each other initially. The first portion is searching for the 21 pallets on the aisles, and both curves exhibit a slight increase in their slope because the travel distance increases as searched pallets are farther from the IO point. This portion is followed by an increase in the steepness of the slopes of the two curves. The increase in the slope of both curves occurs when searching cells that have direct contact with open aisles ends and reposition and then put-back of items is needed to gain access to deeply stored items begins. The slope of the curve representing repositioning policy 2 is greater than that of the curve representing repositioning policy 1, because repositioning pallets outside the store-room requires more distance than repositioning (and put-back) of pallets in the aisles. The curve representing repositioning policy 1 exhibits a third slope increase that coincides with repositioning and put-back locations being farther as the searcher starts progressing in the vertical direction of the storeroom.

To summarize the experiments, Table 1 displays the storeroom configurations varying the accessibility constants at *k = 2, 5, 10, and 20* and length values at a short, medium, and tall length. For repositioning policies 1 and 2, the table provides the values of *k, h,* the store room density, and the corresponding result values of expected search time, the median search time, and the standard deviation of search times. The results are given in time units, with each time unit corresponding to one unit distance.

Table 1: Expected Search Time, Median 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 (*n/N*)	Expected Search Time	Median Search Time	Standard Deviation Search Time	Expected Search Time	Median Search Time	Standard Deviation Search Time
2	6	0.733	24.9	8.5	29.9	27.8	8.5	35.9
2	50	0.792	928.5	52.5	1396.1	1003.3	52.5	1495.0
2	100	0.796	3512.4	102.5	5469.7	3674.3	102.5	5677.5
5	12	0.871	417.2	312.0	397.9	587.1	393.0	586.9
5	50	0.900	5251.3	2920.0	5682.3	6574.9	4076.0	6838.9
5	100	0.905	20896.4	11074.0	23181.7	23780.5	13700.0	25569.8
10	22	0.931	3283.8	2783.0	2805.4	5176.4	4233.5	4526.5
10	50	0.943	13111.4	9516.0	12153.9	19187.4	15115.0	17040.6
10	100	0.948	50726.5	33192.0	50157.5	64735.0	46625.0	60530.6
20	42	0.964	25675.3	22173.0	20683.3	43439.7	37510.0	34998.0
20	100	0.971	112045.4	83058.0	99563.3	168959.1	137136.5	140933.4

From the results displayed in Table 1, both the 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. 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 policy 1 pallets are repositioned to the nearest possible repositioning locations. For small store room configurations, the median search time performance metrics are similar for both repositioning strategies. However, for larger store rooms, repositioning inside open aisles results in median performance being lower than repositioning outside the storeroom. 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 (given the assumptions of our modeling approach) over a policy that uses available space outside the storage area and handles put-back independently of the search process.

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. Searching in a dense storage environment requires travel time, repositioning time and 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 many pallets will require repositioning to gain access to deeply stored pallets, as well as the time needed to put-back the repositioned pallets to their storage locations.

Another insight is that the values of the standard deviation, such as those displayed in Table 1, are large, with many having standard deviations greater than the values of their corresponding expected search time. Therefore, a large amount of variability exists in the values of search times between occupied cells for a given 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 to be found. Such extreme variability contrasts with 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.

Given the high variability, a highlight of our search heuristic is that it provides a discrete probability distribution of search times for all the cells for a given store room configuration. Therefore, 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.

One final insight is that for both repositioning polices, median search times are smaller than the expected search times for all configurations. This indicates that the last cells to be searched contribute higher search values on the expected search time, as visiting those cells require repositioning and put-back (in policy 1) in addition to traveling. The difference between the values of expected search times and median search times signals the existence of a right skewness in the distribution of search times.

To evaluate the performance of the proposed search procedure, we assume the searcher follows a random path while searching for the target item, i.e. starting from the IO location, the searcher will randomly pick an occupied cell location to search and move from there to another random location, and so on. We apply the random search procedure for both repositioning policies, introducing two additional search procedure. *Random Search 1*, in which the searcher randomly chooses the next occupied cell to search while repositioning takes place inside the storeroom, and *Random Search 2*, in which the searcher randomly chooses the next occupied cell to search while repositioning takes place outside the storeroom. We ran each of the two search procedures 100 times for several storeroom configurations. The run with the best (minimum) expected search time out of the 100 runs is compared against the performance of the proposed search procedures to their corresponding random search procedures. Table 2 exhibits the values of the expected search time (EST), median search time (MST), and standard deviation (SD) for each of the four search procedures, for three different storeroom configurations; one small ($k = 2$ and $h = 6$), one medium ($k = 5$ and $h = 50$) and one large ($k = 20$ and $h = 100$).

Table 2: Expected Search Time (EST), Median Search Time (MST) and Standard Deviation of Search Times (SD) in Time Units for repositioning policy 1, random search 1, repositioning policy 2, and random search 2 with different k and h values

	Repositioning Policy 1			**Random Search 1**			**Repositioning Policy 2**			**Random Search 2**		
h	**EST**	**MST**	**SD**	**EST**	**MST**	**SD**	**EST**	**MST**	**SD**	**EST**	**MST**	**SD**
6	24.9	8.5	35.4	60.3	47.0	42.9	27.5	8.5	35.9	53.5	52.0	27
50	5251.3	2920.0	5682.0	16473.8	16851.0	9107.9	6574.9	4076.0	6838.9	15418.0	15399.0	8924
100	112045.4	83058.0	9956.4	284700.0	300588.0	148558.5	168959.1	137137.0	140933.4	292389.0	296597.0	164[illegible]5

From Table 2, the proposed search heuristic outperforms the best performance random search heuristic in both repositioning cases in terms of the expected search time and median search time for all three levels of storeroom configurations. Applying the random search policies resulted in expected and median search time values closer to each other.

9.0 CONCLUSION AND FUTURE RESEARCH

We study a search problem to find a single item in a manual dense storage environment with an inverted T configuration. This is a challenging environment, because some items require being moved out of the way during the search process. Thus, our primary contributions are modeling contributions, as we are the first to study the problem of searching in dense storage, as well as providing a way to break down the problem into the time elements that can be approximated a priori and used as input parameters into an optimization formulation and heuristic procedure.

We formulate a mixed integer linear program to determine the expected search time for a single item in a dense storage system, which is similar to a single machine Job Shop Scheduling Problem (JSSP) with sequence-dependent processing times. We propose a search plan heuristic to solve the optimization problem. 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 during the search process to avoid the impedance of a travel path to the IO 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. 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 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 A repositioning policy that uses the open aisle locations as temporary storage locations and requires put-back of these items while searching is recommended for the environment studied, 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. Finally, when compared to a random search procedures, the proposed search heuristics provide better results in terms of the expected and median search times.

Future extensions can build on both the modeling and the solution approaches in this research. Modeling extensions include 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. Fifth, the optimization formulation can be extended to consider when to conduct put-back as a decision variable. Sixth, more sophisticated solution approaches, such as dynamic programming based approaches, that are more systematic and less myopic can be developed. 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.

ACKNOWLEDGEMENTS

This research is supported by the Office of Naval Research Young Investigator Program: The Design of Responsive Sea-Based Logistic Delivery Systems, Award Number N00014-13-1-0594 and N000141612870.

REFERENCES

Ascheuer, N., Fischetti,M., and Grötschel, M. "Solving the asymmetric travelling salesman problem with time windows by branch-and-cut." *Mathematical Programming* 90.3 (2001): 475-506.

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

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

Bianco, L., Ricciardelli, S., Rinaldi, G., & Sassano, A. (1988). Scheduling tasks with sequence-dependent processing times. *Naval Research Logistics (NRL), 35*(2), 177-184.

Clark, A. V. (2002). *Sea Power 21 Series, Part I: Projecting Decisive Joint Capabilities*. Retrieved from http://www.navy.mil/navydata/cno/proceedings.html

Combined Joint Operations from the Sea Center of Excellence (2012). *An Introduction to Joint Operatiosn on and from the Sea*. Retrieved from http://www.cjoscoe.org/

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

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

Futcher, F. W. (2003). *Selective Offload Capability Simulation (SOCS): An Analysis of High Density Storage Configurations*. Masters Thesis from the Naval Postgraduate School. Retrived 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 IEEE International Conference on Automation Science and Engineering (CASE).

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

Parsons, D. (2013, March). Marine Corps Struggles With Sea-Based Supply Lines. *National Defense*. Retrieved from https://goo.gl/Hvb2ch

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.

Pazour, J.A. & Shin, I. (2016). Logistics Models to Support Order-Fulfillment from the Sea. Progress in Material Handling Research: 2016, Material Handling Institute, Charlotte, NC.

Peters, P (2014). *Startegy for Improving DoD Asset Visibility.* Department of Defense Technical Document Retrieved from https://goo.gl/pfJqFe

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

Reilly, P. J., Pazour, J. A., & Schneider, K. R. (2017). Propagation of unit location uncertainty in dense storage environments. *International Journal of Production Research*, 1-15a.

Scala, N. M., & Pazour, J. A. (2016). A Value Model for Asset Tracking Technology to Support Naval Sea-Based Resupply. *Engineering Management Journal, 28*(2), 120-130.

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

Chapter 7: Batch Order Picking in a Dense Storage System with Item Location Uncertainty

This chapter has been published in the following peer-reviewed journal:

1. Awwad, Mohamed, and Pazour, Jennifer A., 2017, "Batch Order Picking in a Dense Storage System with Item Location Uncertainty," *Proceedings of the 2017 Industrial and Systems Engineering Research Conference, 1619-1624.*

Batch Order Picking in a Dense Storage System with Item Location Uncertainty

We study the problem of picking multiple items in an inverted T k-deep s-stacked dense storage system. The order picker is assumed to be uncertain about exact item locations, therefore the picker will search for items before picking. This dense storage environment is challenging, 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 affected by the number of items stacked over each other. In addition, the order picker is given a task that includes searching and retrieving a batch of items, not just one item. A search heuristic is presented to conduct repositioning, put-back and traveling within a search procedure for a batch of items. We examine a repositioning policy that uses the open aisle locations as temporary storage locations and requires put-back of these items while searching. A set of 80 designed experiments was generated each representing a different dense storeroom configuration and experimental results were analyzed to provide insights.

Keywords

Dense Storage, Search Theory, Greedy Heuristic, Sea-based Logistics

1. Introduction

One of the main objectives of designing a storage system is to use the storage space efficiently while maintaining low labor cost [1]. In a dense storage environment, pallets are stacked horizontally in deep lanes and vertically in high stacks [2]. Dense storage provides a way to increase the storage space utilization by increasing the storage density. A downside of dense storage systems is that some of the stored pallets are not directly accessible. Therefore, the order picker has to move some of the pallets out of the way in order to gain access to other deeply stored pallets in a manually operated dense storage system. A trade-off between labor efficiency and storage density arises in such systems, i.e. a reduction in labor efficiency will result from increasing storage density.

Dense storage systems are used in different applications where storage space is scarce or expensive, such as in city or urban logistics applications, as well as space logistics applications. In this work we are motivated by the US Navy's sea basing concept. A sea base is a floating distribution center that provides supplies to forces deployed ashore participating in various missions in a timely manner. Therefore, a sea base is characterized by its selective offloading capability, which is the sea base's ability to locate and deliver the requested piece of cargo to support forces [3]. For the foreseeable future, sea-based storage systems will be manually operated. Additionally, sea bases will operate in a way resulting in a continuous movement of pallets of cargo stored onboard. Continuous movement of pallets, utilization of a dense storage system and necessity of having the selective offloading capability can result in imperfect visibility of pallets storage locations over the time [4, 5]. The use of asset tracking technologies, such as radio frequency identification (RFID), does not provide the desired granularity of item locations required to eliminate imperfect visibility [6]. Even in traditional aisle-based storage systems, the use of

RFID does not eliminate the need to search for stored items when imperfect knowledge about items' locations exists [7].

Uncertainty about items' location in the storage system adds to the complexity of the manually-operated dense storage environment. The order picker is now forced to search for the group of pallets requested before retrieving them to a designated input/output location in the dense storeroom. Therefore, we are adding a search component to the well-known order picking routing problem. In addition to searching for the requested group of pallets, the order picker or the searcher will have to move pallets out of the way and from on top of each other to gain access to deeply stores pallets.

Inverted T dense storerooms are dense storage systems with one vertical and one horizontal aisle [2]. 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. The letter k will be referred to in this work as the accessibility constant. The value of the accessibility constant is proportional to the accessibility or physical access of any particular pallet in a dense storage system. Stacking s items over each other in the vertical direction increases the storage density in an inverted T system.

As an overview, we are studying the problem of searching for M non-identical items in an inverted T k-deep, s-stacked dense storage system with uncertainty about items' locations. We consider a single order picker or searcher in the system. Our contribution in this research is three-fold:

- Extending the work done in [8] on searching in a dense storage environment by considering a batch of items and increasing the storage density through stacking pallets on top of each other instead of having one level of pallets only stacked on the floor.
- Developing a search plan heuristic for a batch of non-identical items in a dense storage system with an inverted T configuration. The search plan heuristic relies on the use of search time components derived in [9] to calculate the expected search time and standard deviation of search times in various storeroom configurations and sizes.
- Analyzing experimental results and providing insights about the process of searching in manually operated dense storage system with an inverted T configuration. The main purpose of the conducted analysis is detecting the main factors affecting the performance of the proposed search heuristic in terms of the expected search time and standard deviation of search times.

The rest of the paper is organized as follows: In section 2, we review the related literature on dense storage and search theory; in section 3, we provide our problem formulation and assumptions, and; in the following section, we provide the results of the numerical experiments as well as some statistical analysis. Finally, we provide conclusions and suggested future extensions.

2. Literature Review

Two bodies of literature provide the foundation for research in this paper: dense storage and search theory.

2.1. Dense Storage

Over the years, scientists and logisticians have studied ways to increase storage densities, especially in applications where space is scarce or expensive. One way to increase the storage density in a warehouse is dedicating less space for aisles by using narrow or very narrow aisles [10]. Another way is using k-deep pallet racks in which each open location contains k pallets behind each other. In warehouses where k-deep pallet racks are used, each lane is dedicated to pallets from the same Stock-keeping Unit or SKU and there is no need to move a pallet to gain access to another deeply stored one behind it [1]. A third way to increase storage density is through block stacking. In block stacking, pallets from a single SKU are stacked on the floor in a lane that could be several pallets deep and two or more pallets high [11, 12]. Movable aisle systems provide another way of increasing the storage space through placing shelves on sliding carriages to limit unused aisle space [13]. Unlike pallet rack and block stacking storage systems, in the dense storage environment we study stored pallets belong to different SKUs. Therefore, pallets have to be moved from on top of each other or in front of to gain access to lower level or more deeply stored pallets.

In this paper we examine a dense storage system with an inverted T configuration. Inverted T dense storerooms are a special class of the Very High Dense (VHD) storage systems. A VHD storage system is the one in which interfering items exist and need to be moved to gain access to desired items as defined in. When the VHD storage system is automated, it is usually called a deep bulk storage system [2]. The highest storage density for a fixed value of the accessibility constant k is achieved in layouts that resemble the inverted T configuration [3].

To our knowledge, dense storage systems literature assumes order picker's complete knowledge of exact storage locations of all SKUs in a storage system except for the work in [8]. In [8], the order picker has no information about exact SKUs' storage locations and a search heuristic and analytical equations for search time components are derived for an inverted T configuration. It was assumed in [8] that the order picker is picking an order consisting of a single line or SKU and no stacking occurs in the storeroom. However, here we assume that the order picker has to pick a batch of orders each consists of multiple unique SKUs, or a single order consisting of multiple lines or SKUs. The dense storage environment examined in this work also increases the vertical utilization of storage space through stacking pallets on top of each other.

2.2. Search Theory

Search theory was first studied during World War II [15]. In search theory, the problem is to allocate a certain amount of resources to find a target (or a group of targets) whose location is unknown to the searcher, such that a certain measure of effectiveness is maximized. According to the classifications of problems in the search theory literature, the problem studied here falls under discrete stationary target search problems [16]. The search plan here is the dense storage system that is divided into cells. Each cell in the horizontal direction is occupied by one stack of pallets, and each stack in the vertical direction is divided into cells, each of which is occupied by one pallet. Therefore, the problem is considered to be a discrete search problem. The problem is said to be stationary, because we are attempting to find a group of pallets that have a fixed but unknown stationary locations at the moment the order picker or the searcher proceeds with the search procedure. 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.

Searching in an aisle-based storage system has been studied in [7]. They developed a search model based on partially observed Markov Decision Process to study the impact of using a RFID system that exhibits noise to locate an item with an unknown location in a warehouse. They came to the conclusion that searching for an item in a warehouse with the aid of a poorly-designed RFID system would actually cost more than without having the system.

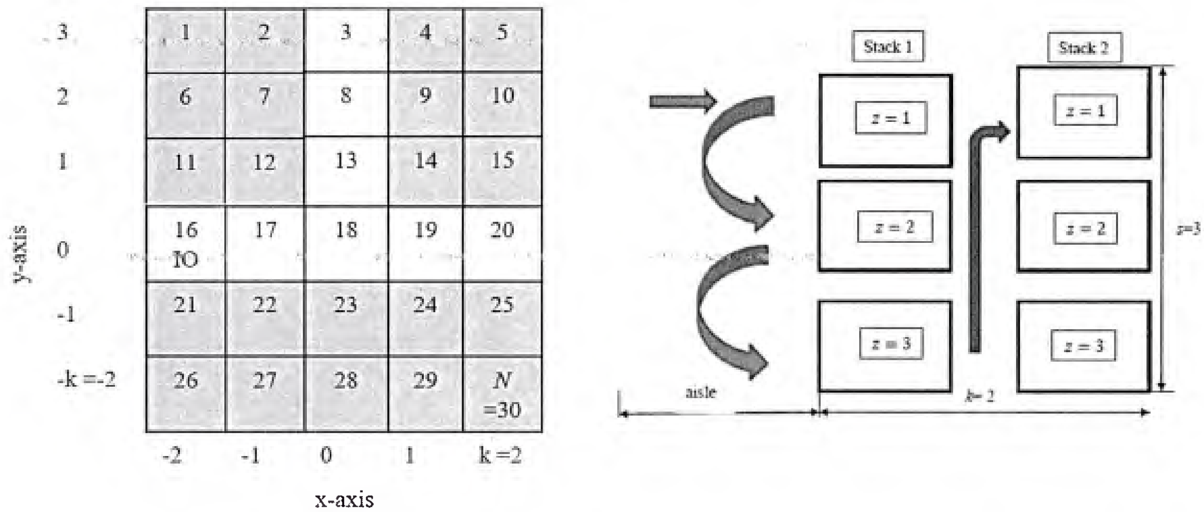
In this work we are studying an inverted T dense storage system that is operated manually. Adding to the complexity of the problem, it is assumed that the order picker does not know the exact locations of stored pallets. A search procedure has to be followed inside the storeroom to find the batch of orders the picker is looking for. The search procedure in dense storage environments requires items to be moved out of the way and from on top of each other while searching. This requires incorporating how to conduct repositioning of pallets to unoccupied locations and put-back movements in an operational search plan.

3. Problem Formulation

A sea base storeroom can be configured as shown in Figure 1. Figure 1a exhibits a 2-D, top view of the inverted T storage system with accessibility constant value $k = 2$. The shown inverted T configuration has one horizontal aisle and one vertical aisle as displayed. Shaded cells represent occupied cells, i.e. cells that contain stacks of pallets. Unshaded cells represent open locations of aisles in which the searcher can travel, and pallets can be repositioned. Figure 1b exhibits an exploded partial side view of the same dense storage system in Figure 1a assuming that there are three pallets stacked over each other, i.e. the stack height $s = 3$.

An x, y, z coordinate system is defined in order to calculate search time analytically. The origin point ($x = 0$ and $y = 0$) is located at the intersection between the vertical and horizontal aisles. The input/output (IO) cell j 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 it reaches $-k$, and increases in the positive direction until the total number of rows is equal to h, where h is the storeroom length. 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. Figure 1 illustrates a 5 by

6 2-deep, 3-stacked inverted T storeroom with coordinates defined for each cell; consequently, $k = 2$, $h = 6$ and $s = 3$.



1a   1b

Figure 1: Sample inverted T system with $k = 2$, $h = 6$ and $s = 3$

As described in more detail in [8], any inverted T storeroom can divided into a number of N cells and stores a number n pallets in it, where $n \leq N$. Each configuration's maximum storage density can be calculated as $\frac{n}{N} = \frac{h(2k+1)-(2k+1)-(h-1-k)}{h(2k+1)}$. In Figure 1, the number of cells is $N = 30$, the maximum number of occupied cells with pallets is $n = 22$, and the associated storage density is 73.3%.

The objective of the search plan is minimizing the expected time required to detect the location of M non-identical items given the absence of knowledge of their exact locations. In this work we assume each item is equally likely to be located in one of the n occupied cells. The search time to find the M items involves 1) the time required to travel between cells through open aisles, 2) the time required to reposition pallets out of the searcher's way and from on top of each other in order to gain access to deeply stored pallets. Pallets are repositioned to other open aisle locations. 3) Finally the time required to put-back repositioned pallets from their temporary locations to their storage locations. Analytical equations for travel and repositioning times are given in [9]. Pallets are put-back in order to maintain the storeroom configuration and to prevent impeding the searcher's way to the IO location once all M pallets are picked. Put-back procedure is initiated if more than k stacks of pallets need to be repositioned and is approximated by put-back time equations.

As an example of the search time components, in Figure 1, if the searcher needs to gain access to the stack of pallets in cell 1 ($x = -2, y = 3$), the stack of pallets located in cell 2 has to be repositioned, one pallet after the other, to open location 2. It is worth noting that when a stack is repositioned to an open aisle location, pallets change their orientation in the vertical direction. If k stacks of pallets have already been repositioned to open aisle locations, the searcher has to initiate the put-back procedure in order to prevent impeding the way to the IO location. For example, if cells 20 and 19 have stacks of pallets repositioned in them, repositioning more pallets in cell 18, for example, will impede the path to the IO location.

The storeroom is represented as a network in which each node represents one of the occupied cells and the lengths of arcs connecting nodes correspond to the search time, which is the sum of travel and repositioning times. When a new batch of orders is generated to pick M items, arcs are generated from the current cell, which is the IO location, to all occupied cells. The cell connected with the arc with the least weight is selected. The index of that cell is added to the search sequence Ψ. This procedure is repeated until the first of M requested items is found, and then this item is retrieved and taken to the IO 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. The search plan heuristic can be summarized through the following pseudo code:

1. Given k, h, s and M.
2. Generate x, y and z coordinates for each pallet in the storeroom.
3. Start with an empty sequence Ψ. Start searching from the IO location and set i = IO cell.
4. Calculate traveling + repositioning time from current cell i to all occupied unvisited cells j as outlined in [9].
5. Select cell j^* with minimum traveling + repositioning time. Break ties arbitrarily. Add cell j^* to the array Ψ.
6. Travel from cell i to the nearest open aisle location to cell j^*.
7. *Reposition* if needed.
8. If j^* is one of the requested M items: retrieve to IO location
   Else if all items are found: stop search and output Ψ

   Else do *put-back* if number repositioned = k,

   Else *set* $i=j^*$ and go to step 4.

4. Numerical Experiments

The output search sequence Ψ using the search plan heuristic from the previous section is used to calculate the search time of each item in the storeroom. For each storeroom configuration, the expected search time (EST) and the standard deviation of search time (SD) are the responses of interest in our statistical analysis. A Design of Experiments approach was used. Three factors that could affect the two responses' values were identified and two levels are set for each of the three factors. A 2^3 full factorial design with 10 replications per run is used. Therefore, a total of 80 instances were randomly generated and each represent a unique configuration. The three factors and their corresponding high and low levels are exhibited in Table 1. An Analysis of Variance (ANOVA) was conducted using Minitab [17] to identify the main and interaction factors that significant effect the expected search time and standard deviation. Table 2 shows the main and interaction effects and the corresponding *p-values* for the expected search time and standard deviation of search times, respectively.

Table 7: Factors and their values at each of the high and low levels

Factor	Low Level (L)	High Level (H)
H/W or the shape factor which is the ratio between the storeroom length h and the storeroom width *w*. The storeroom with the inverted T configuration has a fixed width of $\frac{h}{2k+1}$	k is chosen randomly from uniform distribution U(2,5) with $h = 2k + 2$	k chosen randomly from uniform distribution U(6,10) with $h = 50$
S or the stack height	*s* is chosen randomly from uniform distribution U(1,2)	*s* is chosen randomly from uniform distribution U(3,4)
M or the batch size or the number of non-identical items that need to be picked	*M* is chosen randomly from uniform distribution U(1,5)	*M* is chosen randomly from uniform distribution U(6,10)

From Table 2, at a level of confidence of 95%, the ANOVA results show that only the shape factor, the stack height and the interaction between these two factors statistically have a positive effect on the expected search time. However, the batch size **M** and its interaction with the other two factors do not have a significant effect on the EST value. We can conclude that as the size of the storeroom and the number of pallets stored vertically and horizontally increase, the time needed to find the *M* items will on average increase.

Table 2: Main and interaction effects and corresponding p-values for expected search time and standard deviation

Factors	Effect on EST	*p-value*	Effect on SD	*p-value*
H/W	230524	0.000	78441.6	0.000
S	183722	0.000	62395.6	0.000
M	35539	0.055	-21011.7	0.001
H/W*S	178020	0.000	60332.7	0.000
H/W*M	33993	0.066	-20370.1	0.001
S*M	28158	0.126	-15956.1	0.008
H/W*S*M	26423	0.151	-15556.4	0.010

Table 2 shows as well that three factors and their interactions affect the value of the standard deviation of search times at a significance level of 0.05. However, the main effect of the factor **M** and the effects of its interactions with the other two factors are all negative. Accordingly, as the batch size **M** per search procedure increases the variation in search times, represented by the standard deviation of search times, decreases.

5. Conclusion and Future Research

In this paper, we develop a search plan for a batch of items in a manually operated inverted T dense storage environments with the objective of minimizing the expected time to find all requested items. It is assumed that exact locations of items are unknown to the order picker. Therefore, the searcher has to move pallets out of the way and from on top of each other in order to gain access to deeply stored pallets. Moved pallets are repositioned temporarily inside the storeroom in a way that will not impede the searcher's way to the input/output location upon retrieval. Designed experiments are conducted and an ANOVA approach was used to analyze the results. Results show that as the size of the storeroom and the number of stored pallets increase, the time required to find the requested pallets will increase on average. On the other hand, increasing the batch size decreases the variability in search time for each storeroom configuration. As a future extension, we suggest investigating the use of multiple order pickers instead of just one as we did in this paper, as well as studying other dense storage configurations.

Acknowledgements

This research is supported by the Office of Naval Research Young Investigator Program: The Design of Responsive Sea-Based Logistic Delivery Systems, Award Number N00014-13-1-0594 and N000141612870.

References

1. Bartholdi III, J.J. and S.T. Hackman, *Warehouse & distribution science: release 0.92.* Atlanta, GA, The Supply Chain and Logistics Institute, School of Industrial and Systems Engineering, Georgia Institute of Technology, 2011.

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

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

4. Pazour, J.A. and I. Shin, *Logistics Models to Support Order-Fulfillment from the Sea*, in *Progress in Material Handling Research: 2016.* Material Handling Institute: Charlotte, NC.

5. Reilly, P.J., *Propagation of Unit Location Uncertainty in Dense Storage Environments.* 2015, University of Central Florida.

6. Scala, N.M. and J.A. Pazour, *A Value Model for Asset Tracking Technology to Support Naval Sea-Based Resupply.* Engineering Management Journal, 2016. **28**(2): p. 120-130.

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

8. Awwad, M.A. and J.A. Pazour, *Search Plan for a Single Item in an Inverted T k-Deep Storage System* 2017: Unpublished manuscript.

9. Awwad, M., *Modeling Dense Storage Systems With Location Uncertainty.* 2015, University of Central Florida.

10. Rana, K., *Order picking in narrow-aisle warehouses.* International Journal of Physical Distribution & Logistics Management, 1990. **20**(2): p. 9-15.

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

12. Matson, J. O., J. J. Sonnentag, J. A. White, and R. C. Imhoff, *An Analysis of Block Stacking with Lot Splitting.* in *IIE Annual Conference. Proceedings.* 2014. Institute of Industrial Engineers-Publisher.

13. Tompkins, J.A., and J. D. Smith. *The warehouse management handbook.* Tompkins press, 1998.

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

15. Frost, J. and L.D. Stone, *Review of search theory: advances and applications to search and rescue decision support.* 2001, DTIC Document.

16. Lau, H., S. Huang, and G. Dissanayake. *Optimal search for multiple targets in a built environment.* in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems.* 2005. IEEE.

17. *Minitab 17 Statistical Software.* 2010, Minitab, Inc.: State College, PA.