MULTI-PERIOD EMPTY CONTAINER REPOSITIONING WITH STOCHASTIC DEMAND AND LOST SALES

ZHANG BO

M.Phil

THE HONG KONG POLYTECHNIC UNIVERSITY

2011

The Hong Kong Polytechnic University

Department of Logistics and Maritime Studies

Multi-period Empty Container Repositioning with Stochastic Demand and Lost Sales

ZHANG Bo

A thesis submitted in partial fulfillment of the requirements

for the degree of Master of Philosophy

June 2011

Certificate of Originality

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

______________________________(Signed)

ZHANG Bo (Name of student)

To Mom, Dad and Tian

To the memory of Grandpa

Abstract

This study is concerned with empty container repositioning between multi-ports with stochastic demand and lost sales over multi-periods. Unmet demands due to the unavailability of empty containers will be lost forever and will incur a stockout cost in view of the fact that maritime container shipping is a highly competitive industry. The objective is to find an effective empty container repositioning policy by minimizing the total operating cost including container holding cost, stockout cost, importing cost and exporting cost.

First, this study focuses on the empty container repositioning problem in a single port. This problem is mathematically formulated as an inventory problem over a finite horizon with stochastic import and export of empty containers. The optimal policy for period n is characterized by a pair of critical points (A_n, S_n), i.e., importing empty containers up to A_n when the

number of empty containers in the port is fewer than A_n; exporting empty containers down to S_n when the number of empty containers in the port is more than S_n; and doing nothing, otherwise. A polynomial time algorithm is developed to determine the two thresholds, i.e., A_n and S_n for each period. Two numerical examples are provided to illustrate the solution procedures based on the normal distribution and uniform distribution, respectively. The results show that the proposed algorithm performs highly effectively and efficiently.

Next, this study extends the single-port results to the multi-port case. The multi-port problem is also mathematically formulated and a tight lower bound on the cost function is determined. The concept of relative error with respect to the tight lower bound is introduced, which is used to measure the performance of the algorithm. Based on the two-threshold optimal policy for a single port, a polynomial time algorithm is developed to find an approximate repositioning policy for multi-ports. Simulation results show that the proposed approximate repositioning algorithm performs very effectively as the calculated average relative error with respect to the tight lower bound is within 5 per cent for the normal distribution and uniform distribution, respectively. Furthermore, the algorithm performs very efficiently due to its polynomial running time. The stability of the algorithm improves as the number of ports increases. More importantly, the approximate repositioning policy is easy to understand and implement from a practical perspective as a result of its simplicity.

Acknowledgements

I would like to take this opportunity to express my sincere gratitude to my Chief Supervisor, Dr C.T. Daniel Ng, and my Co-supervisor, Prof. T.C. Edwin Cheng, for their guidance and encouragement throughout my M.Phil study. I really appreciate the efforts made by Dr Ng, who held regular meetings with me every week to inspire me to overcome numerous difficulties in conducting my research, and Prof. Cheng, who proofread my work conscientiously to polish my writing. I cherish the time spent with them and feel honored to be their student.

I would like to thank all the academic staff who gave me lectures or suggestions on my research, and administrative and technical staff for their help. Also many thanks to all the friends I met here, who have enriched my life at PolyU.

I would like to express my deepest appreciation to my parents. I could not have completed this work without their endless love and unceasing support all these years. Finally, I specially thank Tian Lei, who accompanies and supports me all the time. Thank you for your love and care: GTT.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

Movement of goods by sea is the economic lifeblood of many countries. The shipping business has been essential to the development of economic activities as business transactions and trades need ships to transport cargoes from the place of production to the place of consumption. Seaborne shipping has become the major transport mode for international trade since more than 90 per cent of global trade is carried by sea (Lun et al., 2006). The world's seaborne trade has experienced a remarkable increase during the last decade: the volume of seaborne trade has grown from 6,027 million tonnes in 2001 to 8,373 million tonnes in 2010 shown in Figure 1.1.

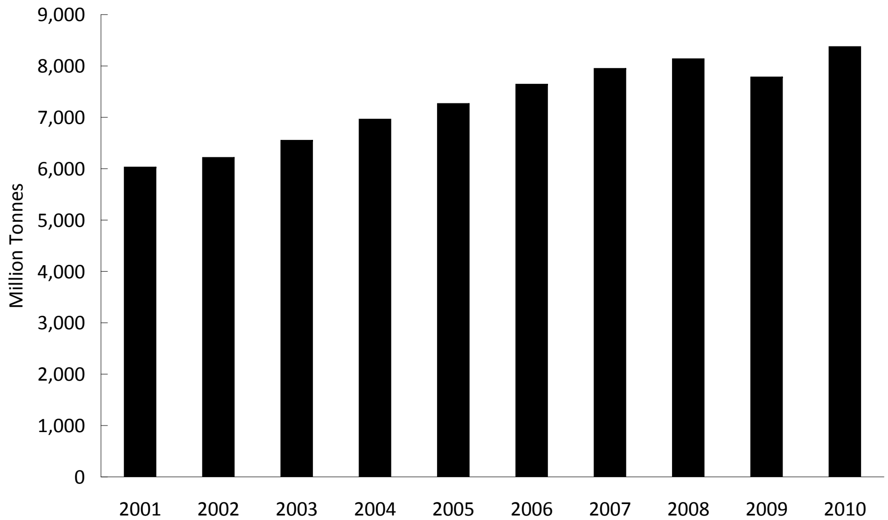

Figure 1.1: World Seaborne Trade

Source: Clarkson (2011b).

However, world seaborne trade has been seriously imbalanced. For example, in the Trans-Pacific shipping route, 15.0 million twenty-foot equivalent units (TEUs) were shipped from Asia to US, while only 4.7 million TEUs were shipped from US to Asia in 2006. Trade imbalance has also happened in other important international shipping routes, such as Asia-Europe route, where 13.5 million TEUs were shipped from Asia to Europe, but only 5.2 million TEUs were shipped from Europe to Asia in 2008. Moreover, the trend for trade imbalance is increasing more and more. The data are summarized in Table 1.1.

Table 1.1: Estimated Cargo Flows on Major Container Trade Routes (Million TEUs)

Year	Trans-Pacific		Asia-Europe	
	Asia-US	US-Asia	Asia-Europe	Europe-Asia
2003	10.2	4.1	7.3	4.9
2004	12.4	4.2	8.9	5.2
2005	12.4	4.4	10.8	5.5
2006	15.0	4.7	15.3	9.1
2007	15.2	5.0	17.2	10.1
2008	13.4	6.9	13.5	5.2
2009	11.5	6.9	11.5	5.5

Source: UNCTAD (2005, 2006, 2007, 2008, 2009, 2010).

Container shipment has become an increasingly popular mode for carrying freight (Dejax and Grainic, 1987). The majority of liner cargo is carried in containers (Lun et al., 2006). Container shipping is estimated to account for more than 70 per cent of world seaborne trade (Drewry, 2005). World container trade has gone through a significant growth over the last decade: the volume of container trade has grown from 68.4 million TEUs in 2001 to 139.8 million TEUs in 2010 as shown in Figure 1.2.

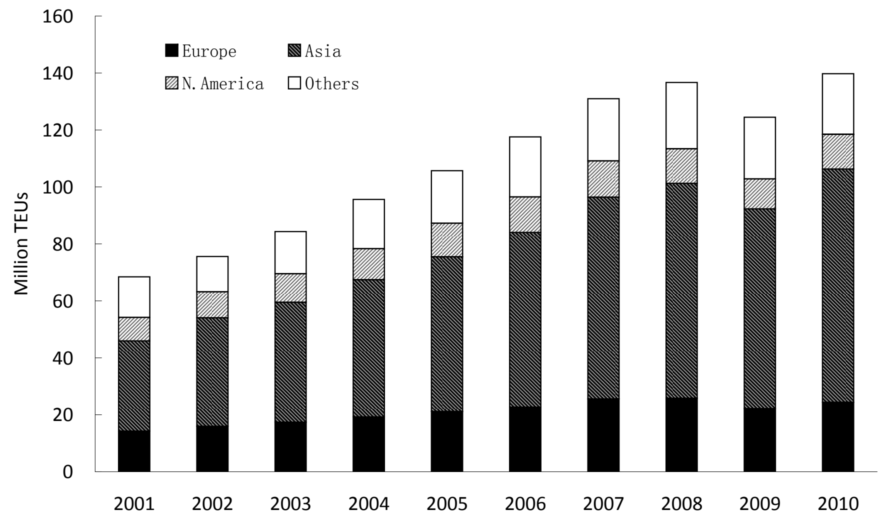

Figure 1.2: World Container Trade

Source: Clarkson (2011a).

In fact, container shipping has been the fastest growing sector of the maritime industries in the last 20 years (Feng and Chang, 2008). Maritime container shipping industry is highly competitive (Cheung and Chen, 1998). A liner shipping company can gain competitive edge by managing its containers effectively to lower the operating costs. Typically, logistics managers' main concern is the transportation of laden containers and they would prefer to ignore empty containers completely, but this is not possible since real-world container networks usually require empty containers to account for the imbalances in loaded flows (Choong et al., 2002). World trade imbalance makes Asia a demanding area for empty containers, and US and Europe surplus areas for empty containers. In order to provide the shippers with the number of containers needed at the right time and place, a liner shipping company has to reposition empty containers from the surplus areas to the deficit areas and bear the container repositioning cost incurred. Song et al. (2005) point out that the repositioning cost for empty containers is 27 per cent of the total world fleet running cost. In this sense, empty container repositioning is a costly and inevitable activity for a liner shipping company to guarantee that the entire shipping networks operate efficiently. It is essential for a liner shipping company to find an effective way to reposition its empty containers. Due to the increasing trade imbalance and randomness of shippers' demands, a liner shipping company is confronted with severe challenges of effective repositioning of empty containers. It is therefore greatly significant to investigate how to reposition empty containers effectively in order to satisfy the shippers' demands and reduce the operating costs in a stochastic and dynamic shipping environment.

This study aims to find an effective empty container repositioning policy to minimize the total operating cost and improve customer satisfaction for a liner shipping company faced with stochastic demand and lost sales.

1.2 Main Contributions

1. To the best of our knowledge, it is the *first* time to solve the empty container repositioning problem in a stochastic and dynamic settings with *lost sales*. It is seen that the lost sales scenario substantially complicates the derivation of the optimal policy mainly due to the max operation on the state variables and others, which results in that the function is *not* differentiable as a whole.

2. The single-port problem is mathematically formulated as an inventory problem over a finite horizon with stochastic import and export of empty containers. The two-threshold *optimal policy* is derived analytically, i.e., for period n: importing empty containers up to A_n when the number of empty containers in the port is fewer than A_n; exporting empty containers down to S_n when the number of empty containers in the port is more than S_n; and doing nothing, otherwise. Furthermore, a polynomial time algorithm is developed to determine the two thresholds A_n and S_n for each period. Two numerical examples are provided to illustrate the solution procedures based on the normal distribution and uniform distribution, respectively. The results show that the proposed algorithm performs effectively and

efficiently.

3. The multi-port problem is mathematically formulated and a tight lower bound (*TLB*) on the cost function is determined. The concept of relative error with respect to the tight lower bound (*RE-TLB*) is introduced, which can be used to measure the performance of a heuristic algorithm. Based on the two-threshold optimal control policy established for a single port, a polynomial time algorithm is developed to find an approximate repositioning policy for multi-ports, which can be easily implemented from a practical perspective. Simulation results show that the proposed algorithm performs very effectively since the average relative error with respect to the tight lower bound (*AVG-RE-TLB*) is within 5 per cent. In addition, the proposed algorithm performs very efficiently due to its polynomial running time.

1.3 Thesis Organization

The rest of this thesis is organized as follows. Chapter 2 presents a comprehensive literature review on empty container repositioning to date. Chapter 3 focuses on the single-port case as the first step towards solving the whole problem. Chapter 4 extends the single-port results to the multi-port case to completely solve the problem. Chapter 5 concludes this study.

Chapter 2

Literature Review

Empty container repositioning problem has aroused great interest of many researchers. Much research has been conducted trying to improve the efficiency of empty container repositioning. Many studies take a deterministic approach. The stochastic characteristics have drawn researchers' attention since the 1990s. Crainic et al. (1993) mathematically formulate the container repositioning and distribution problems, given that demand and supply are stochastic in nature. Crainic and Laporte (1997) identify some of the main issues in freight transportation planning and operations, and present appropriate operations research models, methods and computer based planning tools to address the issues. Cheung and Chen (1998) construct a two-stage stochastic network model to study the dynamic container repositioning problem, whereby they need to reposition empty containers and determine the number of leased containers needed to meet customer demand over time. Lopez (2003) studies the organizational choice of ocean carriers to reposition their empty containers in the USA. Song et al. (2005) propose a formulation for modeling the global container shipping network by considering the cost efficiency and movement patterns. Francesco et al. (2009) propose a time extended multi-scenario optimization model to address a container maritime repositioning problem. Song and Carter (2009) use a mathematical programming approach to evaluate and contrast different strategies that shipping lines and container operators could adopt to reduce the empty container repositioning cost. Theofanis and Boile (2009) summarize several key factors affecting empty container management at global, interregional, regional and local levels. Song and Dong (2011b) propose two types of flow balancing mechanisms for empty container repositioning: one leads to a point-to-point repositioning policy and the other leads to a coordinated repositioning policy. The performance of these two policies is examined in both deterministic and stochastic situations. Many of the above studies apply the mathematical programming approach. However, the mathematical programming models are usually complex and computationally demanding, and the policies are difficult to implement in reality because the underlying logic of these models is hidden from the shipping operator (Du and Hall, 1997).

Plenty of control policies have been proposed recently to address the empty container repositioning issue. Most of them are optimal policies. Li et al. (2004) find two-threshold optimal policies for the single-port empty container repositioning problem under several criteria. Song (2005) investigates the empty vehicle redistribution problem in a two-depot service system with random demands and uncertain transportation times, and determines the optimal stationary policy of threshold control type. Song (2007) studies a periodic-review shuttle service system with random demand and finite repositioning capacity, and characterizes the structures of the optimal stationary policies for both expected discounted cost and long-run average cost. Monotonic and asymptotic behaviors of the optimal policy are also established. Song and Carter (2008) study the empty vehicle redistribution problem in a hub-and-spoke transportation system with random demand and stochastic transportation times. They find an implicit optimal control policy and then present a dynamic decomposition procedure which produces a near-optimal policy. Song and Dong (2008) develop a three-phase threshold control policy to reposition empty containers in cyclic routes, which outperforms the other policies introduced for comparison. Song and Earl (2008) investigate the way to determine optimal control policies for empty vehicle repositioning and fleet-sizing in a two-depot service system with some uncertainties. The optimal empty repositioning policy for a particular fleet size is of threshold control type and the optimal threshold values and fleet-size are also derived. Song et al. (2010) consider the container-dispatching problem in a two-terminal shuttle service system with finite shipping capacity and random customer demand. They derive the optimal policy by using a Markov decision process and propose a well-performed four-parameter threshold policy in an explicit form, which is easy to implement. Song and Zhang (2010) formulate a fluid flow model to analyze empty container repositioning for a single port and use a dynamic programming method to derive the optimal policy in a closed form. Numerous inventory-based control policies are threshold-type, which is easy to understand and implement. In addition, some large shipping lines in UK indicate that they often adopt inventory-based policy to manage their empty containers (Song, 2007).

Simulation methods and heuristic algorithms have attracted wide attention. Lai et al. (1995) establish a simulation model to investigate the policies that yield the lowest operating cost in terms of leasing cost, storage cost, and so on. Lam et al. (2007) apply a simulation based approximate dynamic programming approach to derive an effective operational strategy for a simple two-port two-voyage dynamic container repositioning problem. The modeling and solution approach can be extended to a realistic multiple-port multiple-voyage system. Li et al. (2007) extend the findings of a single-port empty container repositioning problem to multi-port case, and develop a heuristic algorithm to solve this problem. Shintani et al. (2007) develop a genetic algorithm-based heuristics by simultaneously considering the design of container liner shipping service networks and empty container repositioning. Dong and Song (2009) study the joint problem of container fleet sizing and empty container repositioning in multi-vessel, multi-port and multi-voyage shipping systems with dynamic, uncertain and imbalanced customer demand. They use simulation approach based on genetic algorithms and evolutionary strategy. Song and Dong (2011a) consider an empty container repositioning policy with flexible destination ports, which has practical applications in industry. They formulate the policy mathematically and use simulation to evaluate its effectiveness. The results show that it significantly outperforms the conventional policy. Heuristic algorithm combined with simulation method has been shown to be an effective way to deal with the whole complexity of this problem, which lies in the fact that many variables involved in this problem stochastically evolve across the time periods in the planning horizon.

A number of studies involve decision support systems. Shen and Khoong (1995) develop a decision support system to solve the problem of imbalance in supply and demand for empty containers. Cheang and Lim (2005) investigate the dynamic distribution of empty containers, and develop a decision support system to solve the empty container distribution problem. Bandeira et al. (2009) propose a decision support system integrating empty and full containers transshipment operations to address the imbalanced export or import containers repositioning problem, and then use experimental examples to evaluate the effectiveness of the system.

Several studies are on real cases. Choong et al. (2002) computationally analyze the effect of the length of planning horizon on empty container repositioning for intermodal transportation networks, and present a real case. Olivo et al. (2005) propose a mathematical programming method for empty container repositioning, and present a case study of the Mediterranean basin. Feng and Chang (2008) study empty container repositioning planning by considering safety stock management and geographical regions. The problem is treated as a two-stage problem, and case studies of Taiwan Liner Shipping Company are provided to show the applications of the model.

Chapter 3

Single-Port Case

This chapter presents an analysis for the single-port case as the first step to solve the whole empty container repositioning problem for multi-ports. We mathematically formulate the single-port repositioning problem and derive the optimal policy. We also develop a polynomial time algorithm to solve the model, and present two numerical examples to illustrate the solution procedures based on the normal distribution and uniform distribution, respectively.

3.1 Model

This section presents a mathematical formulation of the problem, and derives the optimal policy for a single port.

3.1.1 Notation

1. n: the current discrete decision period,
2. N: the total number of decision periods,
3. i_n: the inventory level of empty containers at the beginning of period n, which is nonnegative, i.e., $i_n \geqslant 0$,
4. d_n: a decision variable, which can be positive (importing empty containers), negative (exporting empty containers), or zero (doing nothing) in period n,
5. u_n: the number of empty containers after making a decision in period n, which is nonnegative, i.e., $u_n = i_n + d_n$ $(u_n \geqslant 0)$,
6. Z_1: the number of laden import containers in each period, which is a nonnegative, independent and identically distributed (i.i.d.) random variable across the decision periods, i.e., $z_1 \geqslant 0$,
7. Z_2: the number of laden export containers in each period, which is a nonnegative, i.i.d. random variable across the decision periods, independent of Z_1, i.e., $z_2 \geqslant 0$,
8. Z: the difference between Z_1 and Z_2, i.e., $Z = Z_1 - Z_2$, which is an i.i.d. random variable across the decision periods,

9. c_h: the holding cost of an empty container per period,

10. c_s: the stockout cost of an empty container per period,

11. c_i: the importing cost of an empty container,

12. c_e: the exporting cost of an empty container,

13. α: the discount factor, $0 < \alpha \leqslant 1$.

In the last item, α is introduced as a result of time value of money, and normally it can be evaluated as $\alpha = (1+r)^{-1}$, where r is the risk-free interest rate for a decision period.

3.1.2 Model Assumptions

1. This is a single commodity model, i.e., all containers are TEU.

2. Leasing policy is *not* considered as an option to supply empty containers in this model.

3. The lost sales scenario is assumed in this model.

4. Laden import containers will become empty containers available for use by the end of each decision period.

5. The process of importing empty containers after making a decision will finish by the end of each decision period.

6. $c_s \geqslant \alpha c_i - c_h$.

In reality, two types of containers, TEU and forty-foot equivalent unit (FEU), are often used (Song, 2007). Because one FEU equals two TEUs (Cheung and Chen, 1998), all containers are assumed to be TEU in this model, i.e., a single commodity model. In fact, Assumption 1 is a standard assumption widely adopted in the literature (e.g., Cheung and Chen, 1998; Choong et al., 2002; Li et al., 2004, 2007; Song, 2007; Song et al., 2010; Song and Zhang, 2010).

A number of previous studies assume that unlimited empty containers can always be leased from leasing companies to avoid empty container unavailability, and containers can be off-leased at any time (e.g., Cheung and Chen, 1998; Li et al., 2004, 2007; Song, 2007; Song and Zhang, 2010). However, such short-term leasing and off-leasing are rather limited in reality, in which shipping companies prefer to lease containers for long-term in order to avoid the expensive short-term leasing (Song et al., 2010). On the other hand, in recent years, the container leasing industry has been shifting towards long-term leasing, and the majority of container leasing firms increasingly place more newly-built containers on long-term leases (Lun et al., 2006). Therefore, no short-term leasing is allowed in this model. Such an assumption can also be found in Song et al. (2010). Moreover, long-term leased containers can be treated as owned containers by a liner shipping company (Cheung and Chen, 1998). Due to the above reasons, leasing policy is not considered as an option to supply empty containers in this model as stated in Assumption 2.

Numerous previous studies assume that all demands *must* be satisfied. If the owned containers are unavailable, unlimited empty containers can always be *leased* from vendors to meet the demands (e.g., Cheung and Chen, 1998; Li et al., 2004, 2007; Song and Zhang, 2010). However, it is probable that *not* all demands can be satisfied in this model as Assumption 2 excludes the leasing policy. Moreover, maritime container shipping is a highly competitive industry (Cheung and Chen, 1998). Therefore, for a liner shipping company, unmet demands due to the unavailability of empty containers will be lost forever and will incur a stockout cost, i.e., lost sales scenario is assumed in this model as stated in Assumption 3.

Some previous studies assume that laden import containers become empty containers available for use immediately after arrivals (e.g., Choong et al., 2002; Song and Zhang, 2010), and the process of importing empty containers will finish immediately after making a decision (e.g., Li et al., 2004, 2007). However, such simplification makes the model barely applicable because it takes some time to complete these two processes in reality, the duration of which cannot be neglected. So our model relaxes their assumptions by only assuming that such processes will finish by the end of each decision period when the ship will sail for other ports, as stated in Assumptions 4 and 5.

The last assumption is indeed a weak condition because stockout will incur not only a loss of profit, but, more importantly, a loss of customer goodwill, which will bring long-term adverse effects on companies. So it is reasonable to set the stockout cost as a large value, which also highlights the significance of lost sales.

3.1.3 Model Formulation

Given that the inventory level of empty containers at the beginning of the current period n is i_n, the number of empty containers available for use after making a decision is $i_n + d_n + Z_1$ due to Assumptions 4 and 5, and Z_2 represents the demand for empty containers. Due to the lost sales scenario in Assumption 3, the inventory level of empty containers at the beginning of the period $n + 1$ will be

$$i_{n+1} = (i_n + d_n + Z_1 - Z_2)^+ = (u_n + Z)^+, \tag{3.1}$$

where $x^+ = \max\{x, 0\}$.

Let $L(u_n)$ be the expected holding cost plus the stockout cost, which can be expressed as follows:

$$L(u_n) = c_h \mathrm{E}(u_n + Z)^+ + c_s \mathrm{E}(-u_n - Z)^+. \tag{3.2}$$

The sum of importing cost and exporting cost is

$$c_i(u_n - i_n)^+ + c_e(i_n - u_n)^+. \tag{3.3}$$

The minimum expected discounted cost starting from period n to the last period N represented by $V_n(i_n)$ satisfies the following recursive relation

$$\begin{aligned} V_n(i_n) = \min_{u_n} \Big\{ & c_i(u_n - i_n)^+ + c_e(i_n - u_n)^+ + L(u_n) \\ & + \alpha \mathrm{E} V_{n+1}(u_n + Z)^+ \Big\}, \end{aligned} \tag{3.4}$$

where $V_{N+1}(i_{N+1}) \equiv 0$ for all i_{N+1}.

3.1.4 Optimal Policy

For convenience, we assume that all the variables are continuous in the proofs below. Generally, the properties for the continuous case also hold for the discrete case. Here we use $f_Z(\cdot)$ and $F_Z(\cdot)$ to represent the probability density function and cumulative distribution function of the random variable Z, respectively. Now define

$$G_n(u_n) = L(u_n) + \alpha \mathrm{E} V_{n+1}(u_n + Z)^+. \tag{3.5}$$

Equivalently,

$$
\begin{aligned}
G_n(u_n) &= L(u_n) + \alpha \mathrm{E} V_{n+1}(u_n+Z)^+ \\
&= c_h \int_{-\infty}^{+\infty} (u_n+z)^+ f_Z(z)\,dz + c_s \int_{-\infty}^{+\infty} (-u_n-z)^+ f_Z(z)\,dz \\
&\quad + \alpha \int_{-\infty}^{+\infty} V_{n+1}(u_n+z)^+ f_Z(z)\,dz \\
&= c_h \int_{-u_n}^{+\infty} (u_n+z) f_Z(z)\,dz + c_s \int_{-\infty}^{-u_n} (-u_n-z) f_Z(z)\,dz \\
&\quad + \alpha \int_{-\infty}^{-u_n} V_{n+1}(0) f_Z(z)\,dz + \alpha \int_{-u_n}^{+\infty} V_{n+1}(u_n+z) f_Z(z)\,dz.
\end{aligned}
\tag{3.6}
$$

According to Leibnitz's rule,

$$
\begin{aligned}
G_n{}'(u_n) &= \frac{dG_n(u_n)}{du_n} \\
&= c_h \int_{-u_n}^{+\infty} f_Z(z)\,dz - c_s \int_{-\infty}^{-u_n} f_Z(z)\,dz \\
&\quad + \alpha \int_{-u_n}^{+\infty} V_{n+1}{}'(u_n+z) f_Z(z)\,dz.
\end{aligned}
\tag{3.7}
$$

Then we define two thresholds, i.e., (A_n, S_n) for period n as follows:

$$A_n = \min\{u_n : G_n{}'(u_n) \geqslant -c_i\}, \tag{3.8}$$

$$S_n = \min\{u_n : G_n{}'(u_n) \geqslant c_e\}. \tag{3.9}$$

Now we are ready to establish Theorem 3.1.

Theorem 3.1. *$V_n(i_n)$ is twice differentiable and convex in i_n for all n and the optimal policy for an n-period problem is characterized by (A_n, S_n).*

Proof. We prove the result by induction. Note that $V_{N+1}(i_{N+1}) \equiv 0$ is twice differentiable and convex in i_{N+1} and ${V_{N+1}}'(i_{N+1}) \geqslant -c_i$. Suppose that ${V_{k+1}}''(i_{k+1}) \geqslant 0$ and ${V_{k+1}}'(i_{k+1}) \geqslant -c_i$. We next prove them for $n = k$.

According to Leibnitz's rule,

$$\begin{aligned}
{G_k}''(u_k) &= \frac{d{G_k}'(u_k)}{du_k} \\
&= (c_h + c_s) f_Z(-u_k) \\
&\quad + \alpha \left[\int_{-u_k}^{+\infty} {V_{k+1}}''(u_k + z) f_Z(z)\, dz + {V_{k+1}}'(0) f_Z(-u_k) \right] \qquad (3.10) \\
&= (\alpha {V_{k+1}}'(0) + c_h + c_s) f_Z(-u_k) \\
&\quad + \alpha \int_{-u_k}^{+\infty} {V_{k+1}}''(u_k + z) f_Z(z)\, dz.
\end{aligned}$$

Due to the induction hypothesis and Assumption 6,

$$\alpha {V_{k+1}}'(0) + c_h + c_s \geqslant -\alpha c_i + c_h + c_s \geqslant 0, \tag{3.11}$$

$${V_{k+1}}''(u_k + z) \geqslant 0. \tag{3.12}$$

Thus,

$${G_k}''(u_k) \geqslant 0. \tag{3.13}$$

So $G_k(u_k)$ is convex and ${G_k}'(u_k)$ is monotone nondecreasing.

Let

$$T_k\left(u_k\right) = c_i(u_k - i_k)^+ + c_e(i_k - u_k)^+ + L\left(u_k\right) + \alpha \mathrm{E} V_{k+1}(u_k + Z)^+. \quad (3.14)$$

Evidently, $T_k\left(u_k\right)$ is continuous.

Equivalently,

$$T_k\left(u_k\right) = \begin{cases} c_e\left(i_k - u_k\right) + G_k\left(u_k\right), & u_k < i_k \\ c_i\left(u_k - i_k\right) + G_k\left(u_k\right), & u_k \geqslant i_k \end{cases}. \quad (3.15)$$

Thus,

$$T_k{}'\left(u_k\right) = \begin{cases} -c_e + G_k{}'\left(u_k\right), & u_k < i_k \\ c_i + G_k{}'\left(u_k\right), & u_k > i_k \end{cases}. \quad (3.16)$$

Case 1: $i_k < A_k$

1. For $u_k < i_k$:

Since $G_k{}'\left(u_k\right) < -c_i$ and $T_k{}'\left(u_k\right) = -c_e + G_k{}'\left(u_k\right) < -c_e - c_i < 0$,

so $T_k\left(u_k\right)$ is monotone decreasing.

2. For $i_k < u_k < A_k$:

Since $G_k{}'\left(u_k\right) < -c_i$ and $T_k{}'\left(u_k\right) = c_i + G_k{}'\left(u_k\right) < 0$,

so $T_k\left(u_k\right)$ is monotone decreasing.

3. For $u_k > A_k$:

Since $G_k{}'\left(u_k\right) \geqslant -c_i$ and $T_k{}'\left(u_k\right) = c_i + G_k{}'\left(u_k\right) \geqslant 0$,

so $T_k\left(u_k\right)$ is monotone nondecreasing.

To summarize, $T_k(u_k)$ is monotone decreasing for $u_k < A_k$ while monotone nondecreasing for $u_k > A_k$.

Since $T_k(u_k)$ is continuous,

$$V_k(i_k) = \min_{u_k}\{T_k(u_k)\} = T_k(A_k) = c_i(A_k - i_k) + G_k(A_k). \tag{3.17}$$

Case 2: $A_k \leqslant i_k \leqslant S_k$

1. For $u_k < i_k$:

Since ${G_k}'(u_k) < c_e$, ${T_k}'(u_k) = -c_e + {G_k}'(u_k) < 0$,

so $T_k(u_k)$ is monotone decreasing.

2. For $u_k > i_k$:

Since ${G_k}'(u_k) \geqslant -c_i$, ${T_k}'(u_k) = c_i + {G_k}'(u_k) \geqslant 0$,

so $T_k(u_k)$ is monotone nondecreasing.

To summarize, $T_k(u_k)$ is monotone decreasing for $u_k < i_k$ while monotone nondecreasing for $u_k > i_k$.

Since $T_k(u_k)$ is continuous,

$$V_k(i_k) = \min_{u_k}\{T_k(u_k)\} = T_k(i_k) = G_k(i_k). \tag{3.18}$$

Case 3: $i_k > S_k$

1. For $u_k < S_k$:

Since ${G_k}'(u_k) < c_e$, ${T_k}'(u_k) = -c_e + {G_k}'(u_k) < 0$,

so $T_k(u_k)$ is monotone decreasing.

2. For $S_k < u_k < i_k$:

Since ${G_k}'(u_k) \geqslant c_e$, ${T_k}'(u_k) = -c_e + {G_k}'(u_k) \geqslant 0$,

so $T_k(u_k)$ is monotone nondecreasing.

3. For $u_k > i_k$:

Since ${G_k}'(u_k) \geqslant c_e$, ${T_k}'(u_k) = c_i + {G_k}'(u_k) \geqslant c_i + c_e > 0$,

so $T_k(u_k)$ is monotone increasing.

To summarize, $T_k(u_k)$ is monotone decreasing for $u_k < S_k$ while monotone nondecreasing for $u_k > S_k$.

Since $T_k(u_k)$ is continuous,

$$V_k(i_k) = \min_{u_k}\{T_k(u_k)\} = T_k(S_k) = c_e(i_k - S_k) + G_k(S_k). \tag{3.19}$$

Combining the above three cases, we find that the optimal policy for period k is characterized by (A_k, S_k), i.e.,

$$V_k(i_k) = \begin{cases} c_i(A_k - i_k) + G_k(A_k), & i_k < A_k \\ G_k(i_k), & A_k \leqslant i_k \leqslant S_k \\ c_e(i_k - S_k) + G_k(S_k), & i_k > S_k \end{cases}. \tag{3.20}$$

Hence,

$$V_k{}'(i_k) = \begin{cases} -c_i, & i_k < A_k \\ G_k{}'(i_k), & A_k \leqslant i_k \leqslant S_k \\ c_e, & i_k > S_k \end{cases} . \tag{3.21}$$

$$V_k{}''(i_k) = \begin{cases} 0, & i_k < A_k \\ G_k{}''(i_k), & A_k \leqslant i_k \leqslant S_k \\ 0, & i_k > S_k \end{cases} . \tag{3.22}$$

Since $V_k{}'(i_k)$ is continuous and $V_k{}''(i_k) \geqslant 0$, $V_k(i_k)$ is convex. The convexity of $V_k(i_k)$ guarantees that $V_k{}'(i_k)$ is monotone nondecreasing, so $V_k{}'(i_k) \geqslant -c_i$. We have shown that $V_k(i_k)$ is convex and $V_k{}'(i_k) \geqslant -c_i$, so the theorem is established. □

According to Theorem 3.1, the optimal policy for period n is characterized by (A_n, S_n), i.e., importing empty containers up to A_n when the number of empty container in the port is fewer than A_n; exporting empty containers down to S_n when the number of empty containers in the port is more than S_n; or doing nothing, otherwise. In other words, if the system is in state i_n at the beginning of period n, the optimal policy $d_n^*(i_n)$ is as follows:

$$d_n^*\left(i_n\right) = \begin{cases} A_n - i_n, & i_n < A_n, \quad \text{import containers to } A_n \\ 0, & A_n \leqslant i_n \leqslant S_n, \quad \text{do nothing} \\ S_n - i_n, & i_n > S_n, \quad \text{export containers to } S_n \end{cases} . \tag{3.23}$$

Propositions 3.1 and 3.2 prove that A_n must exist while under certain conditions S_n does not exist, i.e., $S_n = +\infty$. Thus, two special circumstances need to be clarified here: (1) $A_n = 0$ indicates that optimal policy for period n excludes the possibility of importing empty containers, and (2) $S_n = +\infty$ indicates that optimal policy for period n excludes the possibility of exporting empty containers. By convention, we include such degenerate cases in the optimal policy defined above.

Proposition 3.1. *A_n must exist for all n.*

Proof. We prove the result by contradiction. Suppose that for all u_n, ${G_n}'\left(u_n\right) < -c_i$. Then according to the proof of Theorem 3.1, we know ${V_{n+1}}'\left(i_{n+1}\right) \geqslant -c_i$.

Thus,

$$\begin{aligned}
-c_i > G_n{}'(u_n) &= c_h \int\limits_{-u_n}^{+\infty} f_Z(z)\,dz - c_s \int\limits_{-\infty}^{-u_n} f_Z(z)\,dz \\
&\quad + \alpha \int\limits_{-u_n}^{+\infty} V_{n+1}{}'(u_n+z) f_Z(z)\,dz \\
&\geqslant c_h \int\limits_{-u_n}^{+\infty} f_Z(z)\,dz - c_s \int\limits_{-\infty}^{-u_n} f_Z(z)\,dz \qquad (3.24)\\
&\quad - \alpha c_i \int\limits_{-u_n}^{+\infty} f_Z(z)\,dz \\
&= (c_h - \alpha c_i) \int\limits_{-u_n}^{+\infty} f_Z(z)\,dz - c_s \int\limits_{-\infty}^{-u_n} f_Z(z)\,dz.
\end{aligned}$$

As $u_n \to +\infty$, we have $-c_i \geqslant c_h - \alpha c_i$. And this inequality indicates that $0 \geqslant c_h + (1-\alpha)\, c_i > 0$, which is a contradiction. Thus there exists at least one u_n such that $G_n{}'(u_n) \geqslant -c_i$, so A_n must exist. Furthermore, since n is arbitrary in the above proof, the proposition holds for all n. $\square$

Proposition 3.2. *If $(1-\alpha)\, c_e > c_h$ and $c_i \geqslant c_s$, doing nothing is the optimal policy for all n.*

Proof. $A_n = 0$ and $S_n = +\infty$ for all n is equivalent to this proposition. In Proposition 3.1, we have already shown that

$$\begin{aligned} G_n{}'(u_n) &\geqslant (c_h - \alpha c_i) \int\limits_{-u_n}^{+\infty} f_Z(z)\, dz - c_s \int\limits_{-\infty}^{-u_n} f_Z(z)\, dz \\ &= c_h - \alpha c_i - (c_h + c_s - \alpha c_i) F_Z(-u_n). \end{aligned} \tag{3.25}$$

Similarly, since ${V_{n+1}}'(i_{n+1}) \leqslant c_e$,

$$\begin{aligned} G_n{}'(u_n) &= c_h \int\limits_{-u_n}^{+\infty} f_Z(z)\, dz - c_s \int\limits_{-\infty}^{-u_n} f_Z(z)\, dz \\ &\quad + \alpha \int\limits_{-u_n}^{+\infty} {V_{n+1}}'(u_n + z) f_Z(z)\, dz \\ &\leqslant c_h \int\limits_{-u_n}^{+\infty} f_Z(z)\, dz - c_s \int\limits_{-\infty}^{-u_n} f_Z(z)\, dz \\ &\quad + \alpha c_e \int\limits_{-u_n}^{+\infty} f_Z(z)\, dz \\ &= c_h + \alpha c_e - (c_h + c_s + \alpha c_e) F_Z(-u_n). \end{aligned} \tag{3.26}$$

Since $0 \leqslant F_Z(-u_n) \leqslant 1$, $c_h + c_s - \alpha c_i \geqslant 0$ and $c_h + c_s + \alpha c_e > 0$,

$$G_n{}'(u_n) \geqslant c_h - \alpha c_i - (c_h + c_s - \alpha c_i) F_Z(-u_n) \geqslant -c_s, \tag{3.27}$$

$$G_n{}'(u_n) \leqslant c_h + \alpha c_e - (c_h + c_s + \alpha c_e) F_Z(-u_n) \leqslant c_h + \alpha c_e. \tag{3.28}$$

Thus $-c_s \leqslant G_n{}'(u_n) \leqslant c_h + \alpha c_e$. Under two conditions: $(1-\alpha) c_e > c_h$ and $c_i \geqslant c_s$, we have $-c_i \leqslant -c_s \leqslant G_n{}'(u_n) \leqslant c_h + \alpha c_e < c_e$. Therefore $-c_i \leqslant G_n{}'(u_n) < c_e$, and we obtain $A_n = 0$ and $S_n = +\infty$ based on the definitions of A_n and S_n in Equation 3.8 and 3.9, respectively. Furthermore, since n is arbitrary in the above proof, the proposition holds for all n. $\square$

3.2 Algorithm

We develop a polynomial time algorithm to solve our model. As random variable Z takes discrete values in practice, we need to discretize Z if Z follows a continuous probability distribution. And here we use $p_Z(\cdot)$ to represent the probability mass function (PMF) of the random variable Z after discretization. In reality, random variables Z_1 and Z_2 have an upper bound R, so $0 \leqslant z_1, z_2 \leqslant R$, and similarly, inventory level i has an upper bound M, so $0 \leqslant i \leqslant M$ in Algorithm 3.1.

Before introducing Algorithm 3.1, we first elaborate the way used to calculate the probability mass function of random variable Z and $L(u_n)$, respectively, since they are frequently computed in the algorithm.

Method of Calculating PMF for Random Variable Z

If random variable Z follows a continuous probability distribution, discretize Z:

$$p_Z\left(z\right) \approx \int_{z-0.5}^{z+0.5} f_Z\left(z\right) dz \approx f_Z\left(z\right). \tag{3.29}$$

Else, i.e., random variable Z follows a discrete probability distribution, the probability mass function $p_Z\left(z\right)$ is of itself.

The following method is proposed to calculate $L\left(u_n\right)$, which represents the sum of the expected holding cost and stockout cost. Since $L\left(u_n\right)$ is identical for any period, the subscript n is omitted for brevity.

Method of Calculating $L\left(u\right)$

$$\begin{aligned} L\left(u\right) &= c_h \mathrm{E}(u+Z)^+ + c_s \mathrm{E}(-u-Z)^+ \\ &\approx c_h \sum_{z=-R}^{R} (u+z)^+ p_Z\left(z\right) + c_s \sum_{z=-R}^{R} (-u-z)^+ p_Z\left(z\right). \end{aligned} \tag{3.30}$$

Algorithm 3.1 is proposed to calculate the two thresholds, i.e., A_n and S_n for all periods, i.e., $1 \leqslant n \leqslant N$. This algorithm is developed based on the definition of A_n and S_n in Equations 3.8 and 3.9, respectively, and the monotonicity of the function ${G_n}'\left(\cdot\right)$.

Algorithm 3.1. AS

1. Set $V_{N+1}\left(i\right) = 0$ for all $0 \leqslant i \leqslant M$.

2. Set $n = N$.

3. If $n \geqslant 1$, do the following; else stop.

4. Set $u = -1$.

5. Do this step while ${G_n}'(u) < -c_i$:

   (a) Reset $u = u + 1$.

   (b) Calculate $G_n(u)$:

$$\begin{aligned} G_n(u) &= L(u) + \alpha \mathrm{E} V_{n+1}(u+Z)^+ \\ &\approx L(u) + \alpha \sum_{z=-R}^{R} V_{n+1}(u+z)^+ p_Z(z) \,. \end{aligned} \tag{3.31}$$

   (c) Calculate $G_n(u+1)$:

$$\begin{aligned} G_n(u+1) &= L(u+1) + \alpha \mathrm{E} V_{n+1}(u+1+Z)^+ \\ &\approx L(u+1) + \alpha \sum_{z=-R}^{R} V_{n+1}(u+1+z)^+ p_Z(z) \,. \end{aligned} \tag{3.32}$$

   (d) Calculate ${G_n}'(u)$ approximately:

$${G_n}'(u) \approx \Delta G_n(u) = G_n(u+1) - G_n(u) \,. \tag{3.33}$$

6. Set $A_n = u$.

7. Reset $u = u - 1$.

8. Do this step while ${G_n}'(u) < c_e$:

   (a) Reset $u = u + 1$.

(b) Calculate $G_n(u)$:

$$\begin{aligned} G_n(u) &= L(u) + \alpha \mathrm{E} V_{n+1}(u+Z)^+ \\ &\approx L(u) + \alpha \sum_{z=-R}^{R} V_{n+1}(u+z)^+ p_Z(z) . \end{aligned} \tag{3.34}$$

(c) Calculate $G_n(u+1)$:

$$\begin{aligned} G_n(u+1) &= L(u+1) + \alpha \mathrm{E} V_{n+1}(u+1+Z)^+ \\ &\approx L(u+1) + \alpha \sum_{z=-R}^{R} V_{n+1}(u+1+z)^+ p_Z(z) . \end{aligned} \tag{3.35}$$

(d) Calculate ${G_n}'(u)$ approximately:

$${G_n}'(u) \approx \Delta G_n(u) = G_n(u+1) - G_n(u) . \tag{3.36}$$

9. Set $S_n = u$.

10. Calculate $V_n(i)$ for all $0 \leqslant i \leqslant M$:

$$V_n(i) = \begin{cases} c_i(A_n - i) + G_n(A_n), & i < A_n \\ G_n(i), & A_n \leqslant i \leqslant S_n \\ c_e(i - S_n) + G_n(S_n), & i > S_n \end{cases} . \tag{3.37}$$

11. Reset $n = n - 1$, go to step 3.

Theorem 3.2. *Algorithm 3.1 has time complexity $O(MNR)$.*

Proof. The result is self-evident. □

Theorem 3.2 demonstrates that Algorithm 3.1 is a polynomial time algorithm, which performs very efficiently due to its polynomial running time.

3.3 Numerical Examples

This section presents two numerical examples to illustrate the solution procedures based on the normal distribution and uniform distribution, respectively. The following two examples share the same parameters listed below:

$$M = 1000, \quad R = 50, \quad N = 12, \quad \alpha = 0.99,$$

$$c_h = 180, \quad c_i = 150, \quad c_e = 150, \quad c_s = 1000.$$

The length of the planning horizon is important for empty container repositioning problem. Choong et al. (2002) point out that longer planning horizon allows better management of container movement. In the following two examples, the planning horizon contains 12 consecutive decision periods. When it comes to determining the length of decision period, we need to consider three factors: the time interval between each two consecutive ships, the time to convert laden import containers into empty containers available for use, and the time to import empty containers after making a decision. Normally, each decision period can be approximately one or two weeks. In addition, empty containers can be shipped by different ships sailing on different routes with different shipping schedules since a liner shipping company usually uses the vacancies left after loading the laden containers to reposition the empty containers.

Since the normal distribution and uniform distribution are widely used in the literature (e.g., Song, 2007; Song and Carter, 2009; Song and Dong, 2011b), we use them to illustrate the solution procedures in our examples.

The following results are calculated by running a C++ program on a PC with Pentium 4 CPU 3GHz and 504MB of RAM.

3.3.1 Example I: Normal Distribution

Suppose random variables Z_1 and Z_2 follow the same normal distribution with $\mu_{Z_1} = \mu_{Z_2}$ and $\sigma^2_{Z_1} = \sigma^2_{Z_2} = 50$. Since Z_1 and Z_2 are independent, random variable $Z = Z_1 - Z_2\,(-R \leqslant z \leqslant R)$ follows the normal distribution with $\mu_Z = \mu_{Z_1} - \mu_{Z_2} = 0$ and $\sigma^2_Z = \sigma^2_{Z_1} + \sigma^2_{Z_2} = 100$. We can calculate two thresholds (A_n, S_n) for every period by Algorithm 3.1 and draw the graph of the function $V_1\,(i_1)$. The results are shown in Table 3.1 and Figure 3.1.

Table 3.1: (A_n, S_n) under the Normal Distribution

Period n	1	2	3	4	5	6	7	8	9	10	11	12
A_n	8	8	8	8	8	8	8	8	8	8	9	9
S_n	14	14	14	14	14	14	14	14	14	14	14	16

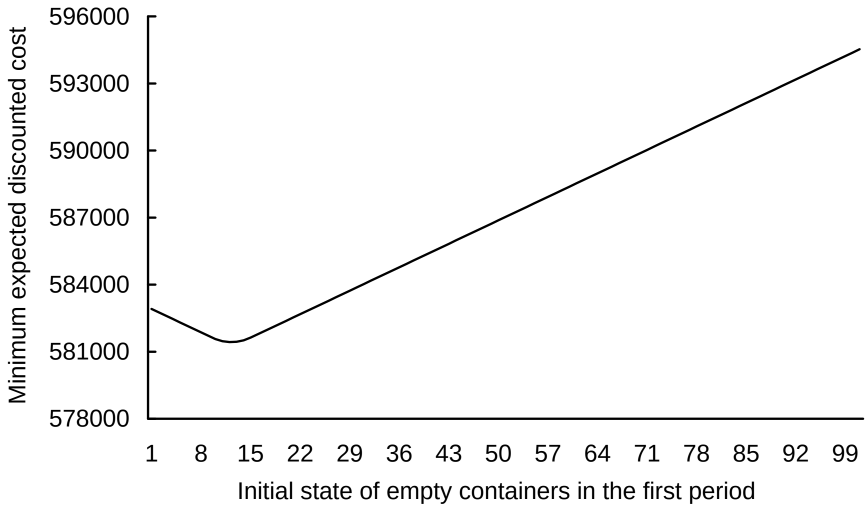

Figure 3.1: Graph of Function $V_1\left(i_1\right)$ under the Normal Distribution

3.3.2 Example II: Discrete Uniform Distribution

Lemma 3.1. *If random variables Z_1 and Z_2 follow the same discrete uniform distribution in the interval $[0, R]$, then the probability mass function of random* *variable $Z = Z_1 - Z_2$ is as follows:*

$$p_Z(z) = \begin{cases} \dfrac{R - |z| + 1}{(R+1)^2}, & -R \leqslant z \leqslant R \\ 0, & \textit{otherwise} \end{cases}. \tag{3.38}$$

Proof. Here we use $p_{Z_1}(\cdot)$ and $p_{Z_2}(\cdot)$ to represent the probability mass function of random variables Z_1 and Z_2, respectively.

For $-R \leqslant z < 0$:

$$\begin{aligned} p_Z(z) &= \sum_{z_1=0}^{R} p_{Z_1}(z_1)\, p_{Z_2}(z_1 - z) \\ &= \frac{1}{R+1} \sum_{z_1=0}^{R+z} p_{Z_2}(z_1 - z) \\ &= \frac{1}{R+1} \frac{R+z+1}{R+1} \\ &= \frac{R+z+1}{(R+1)^2}. \end{aligned} \tag{3.39}$$

For $0 \leqslant z \leqslant R$:

$$\begin{aligned} p_Z(z) &= \sum_{z_1=0}^{R} p_{Z_1}(z_1)\, p_{Z_2}(z_1 - z) \\ &= \frac{1}{R+1} \sum_{z_1=z}^{R} p_{Z_2}(z_1 - z) \\ &= \frac{1}{R+1} \frac{R-z+1}{R+1} \\ &= \frac{R-z+1}{(R+1)^2}. \end{aligned} \tag{3.40}$$

For $z < -R$ or $z > R$:

The fact that Z_1 and Z_2 follow the same discrete uniform distribution in the interval $[0, R]$ means $0 \leqslant z_1, z_2 \leqslant R$, so $-R \leqslant z \leqslant R$. Thus in this case $p_Z(z) = 0$.

The probability mass function of random variable Z is obtained by combining the three cases. Note that the sum of $p_Z(\cdot)$ does equal one. $\square$

Suppose random variables Z_1 and Z_2 follow identical discrete uniform distribution in the interval $[0, R]$. Based on Lemma 3.1, we can calculate two thresholds (A_n, S_n) for every period by Algorithm 3.1 and draw the graph of the function $V_1(i_1)$. The results are shown in Table 3.2 and Figure 3.2.

Table 3.2: (A_n, S_n) under the Discrete Uniform Distribution

Period n	1	2	3	4	5	6	7	8	9	10	11	12
A_n	13	13	13	13	13	13	13	13	13	13	13	13
S_n	31	31	31	31	31	31	31	31	31	31	32	39

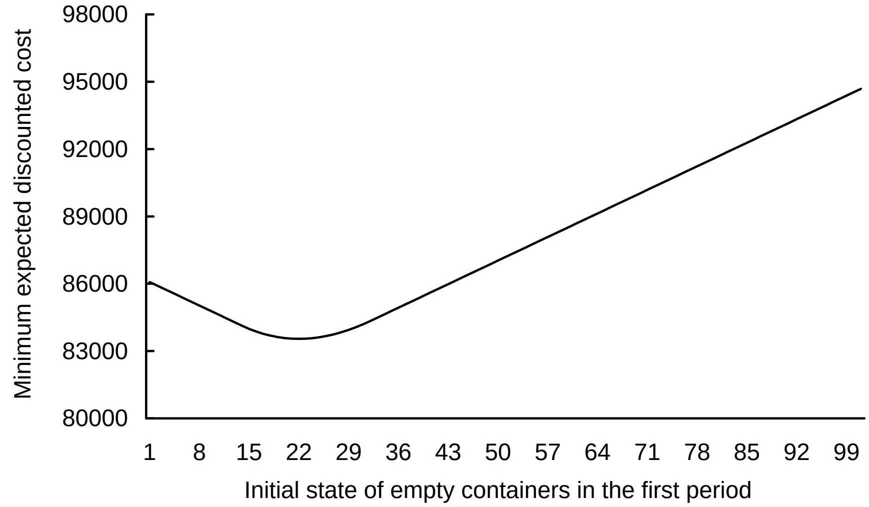

Figure 3.2: Graph of Function $V_1(i_1)$ under the Discrete Uniform Distribution

Figures 3.1 and 3.2 both show that $V_1(i_1)$ is convex in i_1, i.e., the minimum expected discounted cost over the whole planning horizon is a convex function of the initial state of empty containers in the first period, which is consistent with Theorem 3.1.

3.4 Summary

We have formulated the single-port empty container repositioning problem as an inventory problem over a finite horizon with lost sales and stochastic import and export. By solving the stochastic dynamic model, we have found the two-threshold optimal policy. We have also developed a polynomial time algorithm to determine the two thresholds for each period. Then we have presented two numerical examples to illustrate the solution procedures based on the normal distribution and uniform distribution, respectively. The results show that the proposed algorithm performs very effectively and efficiently. The findings in the single-port case contribute to solving the whole repositioning problem as shown in the next chapter.

Chapter 4

Multi-Port Case

This chapter extends the previous single-port case to multi-ports, which becomes much more complicated mainly due to the multi-dimensional variables concerned. We mathematically formulate the multi-port problem, determine a tight lower bound (*TLB*) on the cost function, and introduce the concept of relative error with respect to the tight lower bound (*RE-TLB*). We also develop an approximate polynomial time repositioning algorithm and use simulation to test its performance based on the normal distribution and uniform distribution, respectively.

4.1 Model

This section presents a mathematical formulation of the multi-port problem and a tight lower bound on the cost function. The concept of relative error with respect to the tight lower bound is introduced to evaluate the performance of the proposed algorithm.

4.1.1 Notation

1. n: the current discrete decision period,
2. N: the total number of decision periods,
3. k: port k,
4. K: the total number of ports,
5. e: a vector containing a column of ones,
6. i_n^k: the inventory level of empty containers at the beginning of period n at port k, which is nonnegative, i.e., $i_n^k \geqslant 0$,
7. $\mathbf{i}_n := \left(i_n^1, i_n^2, \cdots, i_n^K\right)' \, (\mathbf{i}_n \geqslant \mathbf{0})$,
8. $I_n := \mathbf{e}'\mathbf{i}_n$, the sum of inventory levels at the beginning of period n over all ports, which is a constant in period n,

9. d_n^k: the decision variable, which can be positive (importing empty containers), negative (exporting empty containers) or zero (doing nothing) in period n at port k,

10. $\mathbf{d}_n := \left(d_n^1, d_n^2, \cdots, d_n^K\right)'$,

11. u_n^k: the number of empty containers after making a decision in period n at port k, which is nonnegative, i.e., $u_n^k = i_n^k + d_n^k \left(u_n^k \geqslant 0\right)$,

12. $\mathbf{u}_n := \left(u_n^1, u_n^2, \cdots, u_n^K\right)'$, $\mathbf{u}_n = \mathbf{i}_n + \mathbf{d}_n \left(\mathbf{u}_n \geqslant \mathbf{0}\right)$,

13. Z_1^k: the number of laden import containers in each period at port k, which is a nonnegative and i.i.d. random variable across the decision periods, i.e., $z_1^k \geqslant 0$,

14. $\mathbf{Z}_1 := \left(Z_1^1, Z_1^2, \cdots, Z_1^K\right)'$,

15. Z_2^k: the number of laden export containers in each period at port k, which is a nonnegative and i.i.d. random variable across the decision periods, independent of Z_1^k, i.e., $z_2^k \geqslant 0$,

16. $\mathbf{Z}_2 := \left(Z_2^1, Z_2^2, \cdots, Z_2^K\right)'$,

17. Z^k: the difference between Z_1^k and Z_2^k, i.e., $Z^k = Z_1^k - Z_2^k$, which is an i.i.d. random variable across the decision periods at port k,

18. $\mathbf{Z} := \left(Z^1, Z^2, \cdots, Z^K\right)' = \mathbf{Z}_1 - \mathbf{Z}_2 = \left(Z_1^1 - Z_2^1, Z_1^2 - Z_2^2, \cdots, Z_1^K - Z_2^K\right)'$,

19. c_h^k: the holding cost of an empty container per period at port k,

20. $\mathbf{c}_h := \left(c_h^1, c_h^2, \cdots, c_h^K\right)'$,

21. c_s^k: the stockout cost of an empty container per period at port k,

22. $\mathbf{c}_s := \left(c_s^1, c_s^2, \cdots, c_s^K\right)'$,

23. c_i^k: the importing cost of empty container per *unit* at port k,

24. $\mathbf{c}_i := \left(c_i^1, c_i^2, \cdots, c_i^K\right)'$,

25. c_e^k: the exporting cost of empty container per *unit* at port k,

26. $\mathbf{c}_e := \left(c_e^1, c_e^2, \cdots, c_e^K\right)'$,

27. α: the discount factor, $0 < \alpha \leqslant 1$.

For items 23 to 26, it is worth noticing that importing and exporting costs are based on *unit*. Generally speaking, shipping one container involves three different costs: lifting-on cost, lifting-off cost and shipping cost from the source to the destination. Since a liner shipping company usually uses the vacancies left after loading the laden containers to reposition the empty containers, the shipping cost of empty containers can be *neglected*, which means only lifting-on cost and lifting-off cost need to be taken into consideration, equal to exporting cost and importing cost, respectively. So the exporting cost and importing cost are on a unit basis. $c_e^p + c_i^q$ represents the total operating cost involved in shipping one empty container from port p to port q.

Definition 4.1. $\mathbf{a} > \mathbf{b}$ *iff* $a_i > b_i$ *for all* $1 \leqslant i \leqslant K$*, where* $\mathbf{a} = (a_1, a_2, \cdots, a_K)'$ *and* $\mathbf{b} = (b_1, b_2, \cdots, b_K)'$.

Definition 4.1 is used to simplify the notation when comparing two vectors.

4.1.2 Model Assumptions

1. This is a single commodity model, i.e., all containers are TEU.

2. Leasing policy is *not* considered as an option to supply empty containers in this model.

3. The lost sales scenario is assumed in this model.

4. Laden import containers will become empty containers available for use by the end of each decision period.

5. The process of importing empty containers after making a decision will finish by the end of each decision period.

6. $\mathbf{c}_s \geqslant \alpha \mathbf{c}_i - \mathbf{c}_h$.

Note that the assumptions here follow or extend those presented in Section 3.1.2, which ensures that the previous results obtained in the single-port case can be applied here to analyze the multi-port case.

Definition 4.2. $\mathbf{a}^+ := \left(a_1^+, a_2^+, \cdots, a_K^+\right)'$, *where* $\mathbf{a} = (a_1, a_2, \cdots, a_K)'$.

Definition 4.2 is used to simplify the notation when formulating the model for multi-ports.

4.1.3 Model Formulation

The system states evolve according to

$$\mathbf{i}_{n+1} = (\mathbf{i}_n + \mathbf{d}_n + \mathbf{Z}_1 - \mathbf{Z}_2)^+ = (\mathbf{u}_n + \mathbf{Z})^+ . \tag{4.1}$$

The sum of the total importing and exporting cost in period n is

$$\mathbf{c}_i{}' (\mathbf{u}_n - \mathbf{i}_n)^+ + \mathbf{c}_e{}' (\mathbf{i}_n - \mathbf{u}_n)^+ , \tag{4.2}$$

Equivalently,

$$\sum_{k=1}^{K} \left[c_i^k \left(u_n^k - i_n^k\right)^+ + c_e^k \left(i_n^k - u_n^k\right)^+ \right] . \tag{4.3}$$

The sum of the total expected holding cost and stockout cost is

$$\sum_{k=1}^{K} L\left(u_n^k\right) . \tag{4.4}$$

To put them together,

$$\sum_{k=1}^{K}\left[c_i^k\left(u_n^k-i_n^k\right)^+ + c_e^k\left(i_n^k-u_n^k\right)^+ + L\left(u_n^k\right)\right]. \tag{4.5}$$

The total minimum expected discounted cost from period n to the last period N represented by $V_n\left(\mathbf{i}_n\right)$ satisfies the following recursive relation

$$\begin{aligned} V_n\left(\mathbf{i}_n\right) = \min_{\mathbf{u}_n}\Bigg\{&\sum_{k=1}^{K}\left[c_i^k\left(u_n^k-i_n^k\right)^+ + c_e^k\left(i_n^k-u_n^k\right)^+ + L\left(u_n^k\right)\right] \\ &+ \alpha \mathrm{E} V_{n+1}\left(\mathbf{u}_n+\mathbf{Z}\right)^+ \Bigg| \mathbf{e}'\mathbf{u}_n = I_n\Bigg\}, \end{aligned} \tag{4.6}$$

where $V_{N+1}\left(\mathbf{i}_{N+1}\right) \equiv 0$ for all $\mathbf{i}_{N+1}$.

4.1.4 Tight Lower Bound

Now we define another similar function $\widetilde{V}_n\left(\mathbf{i}_n\right)$ by removing the constraint $\mathbf{e}'\mathbf{u}_n = I_n$ as follows:

$$\begin{aligned} \widetilde{V}_n\left(\mathbf{i}_n\right) = \min_{\mathbf{u}_n}\Bigg\{&\sum_{k=1}^{K}\left[c_i^k\left(u_n^k-i_n^k\right)^+ + c_e^k\left(i_n^k-u_n^k\right)^+ + L\left(u_n^k\right)\right] \\ &+ \alpha \mathrm{E} \widetilde{V}_{n+1}\left(\mathbf{u}_n+\mathbf{Z}\right)^+ \Bigg\}, \end{aligned} \tag{4.7}$$

where $\widetilde{V}_{N+1}\left(\mathbf{i}_{N+1}\right) \equiv 0$ for all $\mathbf{i}_{N+1}$.

Recall that in the single-port case, for a given port k, $V_n^k\left(i_n^k\right)$ is defined as follows:

$$V_n^k\left(i_n^k\right) = \min_{u_n^k}\left\{c_i^k\left(u_n^k - i_n^k\right)^+ + c_e^k\left(i_n^k - u_n^k\right)^+ + L\left(u_n^k\right)\right.$$
$$\left. + \alpha \mathrm{E} V_{n+1}^k\left(u_n^k + Z^k\right)^+\right\}, \tag{4.8}$$

where $V_{N+1}^k\left(i_{N+1}^k\right) \equiv 0$ for all i_{N+1}^k.

Theorem 4.1. $\widetilde{V}_n\left(\mathbf{i}_n\right) = \sum_{k=1}^K V_n^k\left(i_n^k\right)$ *for all* n*.*

Proof. By induction. Note that $\widetilde{V}_{N+1}\left(\mathbf{i}_{N+1}\right) = \sum_{k=1}^K V_{N+1}^k\left(i_{N+1}^k\right) = 0$. Suppose that $\widetilde{V}_{n+1}\left(\mathbf{i}_{n+1}\right) = \sum_{k=1}^K V_{n+1}^k\left(i_{n+1}^k\right)$, we next prove them for period n. As $\mathbf{i}_{n+1} = \left(\mathbf{u}_n + \mathbf{Z}\right)^+$, where $\mathbf{Z}$ is a random vector, $\mathbf{i}_{n+1}$ is also a random vector. So $\mathrm{E}\widetilde{V}_{n+1}\left(\mathbf{i}_{n+1}\right) = \sum_{k=1}^K \mathrm{E}V_{n+1}^k\left(i_{n+1}^k\right)$, equivalently, $\mathrm{E}\widetilde{V}_{n+1}\left(\mathbf{u}_n + \mathbf{Z}\right)^+ = \sum_{k=1}^K \mathrm{E}V_{n+1}^k\left(u_n^k + Z^k\right)^+$.

Let

$$\widetilde{T}_n\left(\mathbf{u}_n\right) = \sum_{k=1}^K\left[c_i^k\left(u_n^k - i_n^k\right)^+ + c_e^k\left(i_n^k - u_n^k\right)^+ + L\left(u_n^k\right)\right]$$
$$+ \alpha \mathrm{E}\widetilde{V}_{n+1}\left(\mathbf{u}_n + \mathbf{Z}\right)^+. \tag{4.9}$$

Due to

$$T_n^k\left(u_n^k\right) = c_i^k\left(u_n^k - i_n^k\right)^+ + c_e^k\left(i_n^k - u_n^k\right)^+ + L\left(u_n^k\right)$$
$$+ \alpha \mathrm{E}V_{n+1}^k\left(u_n^k + Z^k\right)^+, \tag{4.10}$$

Thus,

$$
\begin{aligned}
\widetilde{T}_n(\mathbf{u}_n) &= \sum_{k=1}^{K}\left[c_i^k\left(u_n^k - i_n^k\right)^+ + c_e^k\left(i_n^k - u_n^k\right)^+ + L\left(u_n^k\right)\right] \\
&\quad + \sum_{k=1}^{K}\alpha \mathrm{E} V_{n+1}^k\left(u_n^k + Z^k\right)^+ \\
&= \sum_{k=1}^{K}\Big[c_i^k\left(u_n^k - i_n^k\right)^+ + c_e^k\left(i_n^k - u_n^k\right)^+ + L\left(u_n^k\right) \\
&\quad + \alpha \mathrm{E} V_{n+1}^k\left(u_n^k + Z^k\right)^+\Big] \\
&= \sum_{k=1}^{K} T_n^k\left(u_n^k\right).
\end{aligned}
\tag{4.11}
$$

So $\min_{\mathbf{u}_n}\left\{\widetilde{T}_n(\mathbf{u}_n)\right\} = \min_{\mathbf{u}_n}\left\{\sum_{k=1}^{K} T_n^k\left(u_n^k\right)\right\} = \sum_{k=1}^{K}\min_{u_n^k}\left\{T_n^k\left(u_n^k\right)\right\}$, where the second equality is based on the independence of the elements of $\mathbf{u}_n$. Since $\widetilde{V}_n(\mathbf{i}_n) = \min_{\mathbf{u}_n}\left\{\widetilde{T}_n(\mathbf{u}_n)\right\}$ and $V_n^k\left(i_n^k\right) = \min_{u_n^k}\left\{T_n^k\left(u_n^k\right)\right\}$, therefore, $\widetilde{V}_n(\mathbf{i}_n) = \sum_{k=1}^{K} V_n^k\left(i_n^k\right)$. $\square$

Let $\mathbf{u}_n^*(\mathbf{i}_n)$ be the optimal $\mathbf{u}_n$ corresponding to $\widetilde{V}_n(\mathbf{i}_n)$, and $u_n^{k*}\left(i_n^k\right)$ be the optimal u_n^k corresponding to $V_n^k\left(i_n^k\right)$. As in the real situation, all possible values for $\mathbf{u}_n$ are finite, thus $\mathbf{u}_n^*$ must exist. Theorem 4.1 shows that $\mathbf{u}_n^*$ is the collection of u_n^{k*} for all n, i.e., $\mathbf{u}_n^* = \left(u_n^{1*}, u_n^{2*}, \cdots, u_n^{K*}\right)'$ for all n.

Theorem 4.2. *For all n, $V_n(\mathbf{i}_n) \geqslant \widetilde{V}_n(\mathbf{i}_n)$ and equality holds if $(1-\alpha)\,\mathbf{c}_e > \mathbf{c}_h$ and $\mathbf{c}_i \geqslant \mathbf{c}_s$.*

Proof. According to the optimization theory, the constrained minimum cannot be less than the unconstrained minimum, i.e., $V_n(\mathbf{i}_n) \geqslant \widetilde{V}_n(\mathbf{i}_n)$. According to Proposition 3.2, if $(1-\alpha)\,\mathbf{c}_e > \mathbf{c}_h$ and $\mathbf{c}_i \geqslant \mathbf{c}_s$, doing nothing is the optimal policy for all ports in any period, so $u_n^{k*} = i_n^k$ for all k, thus $\mathbf{u}_n^* = \mathbf{i}_n$ due to $\mathbf{u}_n^* = \left(u_n^{1*}, u_n^{2*}, \cdots, u_n^{K*}\right)'$. Therefore, $\mathbf{e}'\mathbf{u}_n^* = \mathbf{e}'\mathbf{i}_n = I_n$, so the constraint $\mathbf{e}'\mathbf{u}_n = I_n$ is satisfied, which means $V_n(\mathbf{i}_n) = \widetilde{V}_n(\mathbf{i}_n)$ in this case. Futhermore, since n is arbitrary in the above proof, the theorem holds for all n. $\square$

Theorem 4.3. *$\sum_{k=1}^{K} V_n^k\left(i_n^k\right)$ is a tight lower bound on $V_n(\mathbf{i}_n)$ for all n.*

Proof. Due to Theorems 4.1 and 4.2, $V_n(\mathbf{i}_n) \geqslant \sum_{k=1}^{K} V_n^k\left(i_n^k\right)$ and equality holds if $(1-\alpha)\,\mathbf{c}_e > \mathbf{c}_h$ and $\mathbf{c}_i \geqslant \mathbf{c}_s$. Furthermore, since n is arbitrary in the above proof, the theorem holds for all n. $\square$

4.1.5 Relative Error with respect to the Tight Lower Bound

The relative error is used to measure the extent to which the cost given by the heuristic algorithm deviates from the optimal value, which is dimensionless, making it convenient to compare. Analogically, the relative error with respect to the tight lower bound is introduced to measure the extent to which the cost given by the heuristic algorithm deviates from the corresponding tight lower bound in this study.

Lemma 4.1. *For a given problem instance, the relative error is less than or equal to the relative error with respect to the tight lower bound for positive minimization problem.*

Proof. Let I be a given problem instance, $H(I)$ be the value obtained by the heuristic algorithm, $M(I)$ be the theoretical *minimum* value, $TLB(I)$ be the tight lower bound on $M(I)$, and suppose $H(I)$, $M(I)$ and $TLB(I)$ are all *positive* values. Let $RE(I)$ be the relative error. Given the problem instance I, $RE(I)$ is defined as $RE(I) := \frac{H(I)-M(I)}{M(I)} = \frac{H(I)}{M(I)} - 1$ and $RE\text{-}TLB(I) := \frac{H(I)-TLB(I)}{TLB(I)} = \frac{H(I)}{TLB(I)} - 1$. Due to $M(I) \geqslant TLB(I) > 0$ and $H(I) > 0$, so $RE(I) \leqslant RE\text{-}TLB(I)$. $\square$

Lemma 4.1 provides another possible way to evaluate the performance of a heuristic algorithm. In the situation when the theoretical minimum value is not easy or even impossible to calculate but a tight lower bound is determined, we are still able to draw the conclusion that the heuristic algorithm performs effectively as long as the calculated relative error with respect to the tight lower bound is sufficiently small, say, less than 5 per cent. It is because the underlying true relative error must be identical or even smaller according to Lemma 4.1.

4.2 Algorithm

Since it is extremely difficult to derive optimal policy in the multi-port case, a polynomial time algorithm is developed to find an approximate repositioning policy.

Suppose we have already calculated A_n^k and S_n^k for all $1 \leqslant n \leqslant N$ and $1 \leqslant k \leqslant K$ beforehand. We treat this multi-stage decision problem as a sequence of several single-stage decision problems, and make a decision at the beginning of each decision period sequentially when i_n^k for every port is realized. Without loss of generality, assume that we are in the n-th period, based on $\left(A_n^k, S_n^k\right)$ for port k, we define three state sets as follows:

$$\Omega_n^a := \left\{k | i_n^k < A_n^k\right\} \quad \Omega_n^b := \left\{k | A_n^k \leqslant i_n^k \leqslant S_n^k\right\} \quad \Omega_n^c := \left\{k | i_n^k > S_n^k\right\}$$

It is clear that in every period, each port must fall into only one of the three sets, and all ports therefore are classified into three different groups. According to the optimal policy established in Theorem 3.1, in each period, importing empty containers for $k \in \Omega_n^a$ and exporting empty containers for $k \in \Omega_n^c$ can reduce the total operating cost, while either importing or exporting empty containers for $k \in \Omega_n^b$ will add to the total operating cost.

For any port k, importing one empty container will change the total operating cost by

$$\overline{\Delta}_n^k\left(i_n^k\right) = G_n^k\left(i_n^k + 1\right) - G_n^k\left(i_n^k\right) + c_i^k \tag{4.12}$$

For any port k, exporting one empty container will change the total operating cost by

$$\underline{\Delta}_n^k\left(i_n^k\right) = G_n^k\left(i_n^k - 1\right) - G_n^k\left(i_n^k\right) + c_e^k \tag{4.13}$$

Note that

$$\text{For } k \in \Omega_n^a: \quad \overline{\Delta}_n^k\left(i_n^k\right) \leqslant 0, \quad \underline{\Delta}_n^k\left(i_n^k\right) > 0. \tag{4.14}$$

$$\text{For } k \in \Omega_n^b: \quad \overline{\Delta}_n^k\left(i_n^k\right) > 0, \quad \underline{\Delta}_n^k\left(i_n^k\right) > 0. \tag{4.15}$$

$$\text{For } k \in \Omega_n^c: \quad \overline{\Delta}_n^k\left(i_n^k\right) > 0, \quad \underline{\Delta}_n^k\left(i_n^k\right) \leqslant 0. \tag{4.16}$$

For any port k, since $c_s^k > \alpha c_i^k - c_h^k$ is almost true in reality, $G_n^k\left(\cdot\right)$ is strictly convex and the above inequalities hold strictly.

According to the optimal policy established in Theorem 3.1, it would *never* be optimal, in any circumstance, to export empty containers for $k \in \Omega_n^a$ or import empty containers for $k \in \Omega_n^c$. Therefore, we re-define *Delta* used in Algorithm 4.1 as follows:

$$\overline{\Delta}_n^k\left(i_n^k\right) = \begin{cases} G_n^k\left(i_n^k + 1\right) - G_n^k\left(i_n^k\right) + c_i^k, & i_n^k < S_n^k \\ \text{a huge number}, & i_n^k \geqslant S_n^k \end{cases}. \tag{4.17}$$

$$\underline{\Delta}_n^k\left(i_n^k\right) = \begin{cases} G_n^k\left(i_n^k - 1\right) - G_n^k\left(i_n^k\right) + c_e^k, & i_n^k > A_n^k \\ \text{a huge number}, & i_n^k \leqslant A_n^k \end{cases}. \tag{4.18}$$

Algorithm 4.1 is proposed to reposition the empty containers between multi-ports. We have calculated the A_n^k and S_n^k for the whole planning horizon, i.e., $1 \leqslant n \leqslant N$ and all the ports involved, i.e., $1 \leqslant k \leqslant K$ beforehand by Algorithm 3.1. Since Algorithm 4.1 is applicable to any period, we omit the subscript n for brevity.

Algorithm 4.1. Reposition

1. For each k within $1 \leqslant k \leqslant K$:
   (a) If $i^k < A^k$, add k to state set Ω^a.
   (b) If $A^k \leqslant i^k \leqslant S^k$, add k to state set Ω^b.
   (c) If $i^k > S^k$, add k to state set Ω^c.
2. (a) If $\Omega^a = \emptyset$ and $\Omega^c = \emptyset$, stop.
   (b) If $\Omega^a \neq \emptyset$ and $\Omega^c \neq \emptyset$, go to step 3.
   (c) If $\Omega^a = \emptyset$ and $\Omega^c \neq \emptyset$,

      If $\Omega^b \neq \emptyset$, go to step 4; else stop.

(d) If $\Omega^a \neq \emptyset$ and $\Omega^c = \emptyset$,

If $\Omega^b \neq \emptyset$, go to step 5; else stop.

3. (a) Find the p with the minimum $\overline{\Delta}^k\left(i^k\right)$ from $k \in \Omega^a$.

   (b) Fine the q with the minimum $\underline{\Delta}^k\left(i^k\right)$ from $k \in \Omega^c$.

   (c) Reset $i^p = i^p + 1$.

   (d) Reset $i^q = i^q - 1$.

   (e) If $i^p = A^p$,

   - Delete p from Ω^a;
   - Add p to Ω^b.

   (f) If $i^q = S^q$,

   - Delete q from Ω^c;
   - Add q to Ω^b.

   (g) Go to step 2.

4. (a) Find the p with the minimum $\underline{\Delta}^k\left(i^k\right)$ from $k \in \Omega^c$.

   (b) Find the q with the minimum $\overline{\Delta}^k\left(i^k\right)$ from $k \in \Omega^b$.

   (c) If $\underline{\Delta}^p\left(i^p\right) + \overline{\Delta}^q\left(i^q\right) \geqslant 0$, stop; else

   - Reset $i^p = i^p - 1$.
   - Reset $i^q = i^q + 1$.

- If $i^p = S^p$, delete p from Ω^c.
- If $i^q = S^q$, delete q from Ω^b.
- Go to step 2.

5. (a) Find the p with the minimum $\overline{\Delta}^k\left(i^k\right)$ from $k \in \Omega^a$.

   (b) Fine the q with the minimum $\underline{\Delta}^k\left(i^k\right)$ from $k \in \Omega^b$.

   (c) If $\overline{\Delta}^p\left(i^p\right) + \underline{\Delta}^q\left(i^q\right) \geqslant 0$, stop; else

   - Reset $i^p = i^p + 1$.
   - Reset $i^q = i^q - 1$.
   - If $i^p = A^p$, delete p from Ω^a.
   - If $i^q = A^q$, delete q from Ω^b.
   - Go to step 2.

Theorem 4.4. *Algorithm 4.1 has time complexity $O\left(K\right)$.*

Proof. The result is self-evident. □

Theorem 4.4 demonstrates that Algorithm 4.1 is a polynomial time algorithm, which performs very efficiently due to its polynomial running time.

Theorem 4.5. *The overall time complexity for solving the whole empty container repositioning problem is $O\left(KMNR\right)$.*

Proof. Since Algorithm 4.1 is executed at the beginning of each decision period when the inventory levels for all the ports are realized, the time complexity corresponding to the whole planning horizon containing N consecutive decision periods is $O(KN)$. Moreover, Algorithm 3.1 is executed K times beforehand to calculate the A_n^k and S_n^k for all the ports over the whole planning horizon. Thus, the corresponding time complexity is $O(KMNR)$. Because $\max\{O(KN), O(KMNR)\} = O(KMNR)$, the overall time complexity to solve the whole problem is $O(KMNR)$. □

Theorem 4.5 demonstrates that the whole empty container repositioning problem can be solved within a polynomial running time, which reflects the high efficiency of the proposed Algorithms 3.1 and 4.1.

4.3 Simulation

This section presents the simulation procedures and evaluation of the performance of Algorithm 4.1 under the normal distribution and uniform distribution, respectively.

4.3.1 Simulation Procedures

Three subprocedures are proposed first since they are frequently called by the simulation procedures.

Subprocedure 4.1 is used to reposition the empty containers and calculate the corresponding operating costs in the simulation. The main part follows from Algorithm 4.1. To generate random number Z, we first generate a pseudorandom number uniformly distributed over $(0, 1)$, then generate Z using the *inverse transform method* for the discrete uniform distribution and the *rejection method* for the normal distribution (Ross, 2006). In practice, inventory level i^k has an upper bound M, i.e., $0 \leqslant i^k \leqslant M$ for all k and random variables Z_1^k and Z_2^k have an upper bound R, i.e., $0 \leqslant z_1^k, z_2^k \leqslant R$ for all k in Subprocedure 4.1.

Subprocedure 4.1. Reposition

1. Set $n = 1$.
2. If $n \leqslant N$, do the following; else stop.
3. For each k within $1 \leqslant k \leqslant K$:
   (a) If $i^k < A^k$,
      - Add k to state set Ω^a;

- Set $u^k = A^k$, according to the optimal policy;
- Reset $Importing^{TLB}[k] = Importing^{TLB}[k] + c_i^k \left(u^k - i^k\right)$, correspondingly.

(b) If $A^k \leqslant i^k \leqslant S^k$,

- Add k to state set Ω^b;
- Set $u^k = i^k$, according to the optimal policy.

(c) If $i^k > S^k$,

- Add k to state set Ω^c;
- Set $u^k = S^k$, according to the optimal policy;
- Reset $Exporting^{TLB}[k] = Exporting^{TLB}[k] + c_e^k \left(i^k - u^k\right)$, correspondingly.

4. (a) If $\Omega^a = \emptyset$ and $\Omega^c = \emptyset$, go to step 8.

   (b) If $\Omega^a \neq \emptyset$ and $\Omega^c \neq \emptyset$, go to step 5.

   (c) If $\Omega^a = \emptyset$ and $\Omega^c \neq \emptyset$,

   If $\Omega^b \neq \emptyset$, go to step 6; else go to step 8.

   (d) If $\Omega^a \neq \emptyset$ and $\Omega^c = \emptyset$,

   If $\Omega^b \neq \emptyset$, go to step 7; else go to step 8.

5. (a) Find the p with the minimum $\overline{\Delta}^k \left(i^k\right)$ from $k \in \Omega^a$.

(b) Fine the q with the minimum $\underline{\Delta}^k \left(i^k\right)$ from $k \in \Omega^c$.

(c) Reset $i^p = i^p + 1$.

(d) Reset $i^q = i^q - 1$.

(e) Reset $Exporting\,[q] = Exporting\,[q] + c_e^q$, correspondingly.

(f) Reset $Importing\,[p] = Importing\,[p] + c_i^p$, correspondingly.

(g) If $i^p = A^p$,

- Delete p from Ω^a;
- Add p to Ω^b.

(h) If $i^q = S^q$,

- Delete q from Ω^c;
- Add q to Ω^b.

(i) Go to step 4.

6. (a) Find the p with the minimum $\underline{\Delta}^k \left(i^k\right)$ from $k \in \Omega^c$.

(b) Find the q with the minimum $\overline{\Delta}^k \left(i^k\right)$ from $k \in \Omega^b$.

(c) If $\underline{\Delta}^p \left(i^p\right) + \overline{\Delta}^q \left(i^q\right) \geqslant 0$, go to step 8; else

- Reset $i^p = i^p - 1$.
- Reset $i^q = i^q + 1$.
- Reset $Exporting\,[p] = Exporting\,[p] + c_e^p$, correspondingly.

- Reset $Importing\left[q\right] = Importing\left[q\right] + c_i^q$, correspondingly.
- If $i^p = S^p$, delete p from Ω^c.
- If $i^q = S^q$, delete q from Ω^b.
- Go to step 4.

7. (a) Find the p with the minimum $\overline{\Delta}^k\left(i^k\right)$ from $k \in \Omega^a$.

   (b) Fine the q with the minimum $\underline{\Delta}^k\left(i^k\right)$ from $k \in \Omega^b$.

   (c) If $\overline{\Delta}^p\left(i^p\right) + \underline{\Delta}^q\left(i^q\right) \geqslant 0$, go to step 8; else

   - Reset $i^p = i^p + 1$.
   - Reset $i^q = i^q - 1$.
   - Reset $Exporting\left[q\right] = Exporting\left[q\right] + c_e^q$, correspondingly.
   - Reset $Importing\left[p\right] = Importing\left[p\right] + c_i^p$, correspondingly.
   - If $i^p = A^p$, delete p from Ω^a.
   - If $i^q = A^q$, delete q from Ω^b.
   - Go to step 4.

8. For each k within $1 \leqslant k \leqslant K$:

   (a) Generate a random number Z.

   (b) If $i^k + Z \geqslant 0$,

   Reset $Holding\left[k\right] = Holding\left[k\right] + c_h^k \min\{i^k + Z, M\}$;

Else

Reset $Stockout\,[k] = Stockout\,[k] - c_s^k\left(i^k + Z\right)$.

(c) If $u^k + Z \geqslant 0$,

Reset $Holding^{TLB}\,[k] = Holding^{TLB}\,[k] + c_h^k \min\{u^k + Z, M\}$;

Else

Reset $Stockout^{TLB}\,[k] = Stockout^{TLB}\,[k] - c_s^k\left(u^k + Z\right)$.

(d) Reset $i^k = \min\left\{\left(i^k + Z\right)^+, M\right\}$.

9. Clear Ω^a, Ω^b, Ω^c.

10. Reset $n = n + 1$, go to step 2.

Lemma 4.2. $U' = a + (b - a)\,U \sim U\,(a, b)$ *if* $U \sim U\,(0, 1)$.

Proof. The result is self-evident. □

Lemma 4.2 is a well-known result, which is usually used to generate the desired uniform variable from the standard uniform distribution in simulation.

Subprocedure 4.2 is used to initialize the problem instance, which is determined by the combination of 8 parameters $\mathbf{c}_i$, $\mathbf{c}_e$, $\mathbf{c}_h$, $\mathbf{c}_s$, $\mathbf{i}_1$, M, N and

R, in which $\mathbf{c}_i$, $\mathbf{c}_e$, $\mathbf{c}_h$, $\mathbf{c}_s$ and $\mathbf{i}_1$ are all vectors. The elements of each vector are all randomly generated from a given uniform distribution in order to better test the effectiveness of Algorithm 4.1. The supports of the uniform distributions are defined by two preassigned values, respectively, i.e., (c_i^{min}, c_i^{max}), (c_e^{min}, c_e^{max}), (c_h^{min}, c_h^{max}), (c_s^{min}, c_s^{max}) and (i^{min}, i^{max}). Lemma 4.2 is applied to conduct the transformation of uniform variables.

Subprocedure 4.2. Initialization

For each k within $1 \leqslant k \leqslant K$:

1. (a) Generate a pseudorandom number U uniformly distributed over $(0, 1)$.

   (b) Set $c_i^k = c_i^{min} + (c_i^{max} - c_i^{min})\,U$ to make c_i^k uniformly distributed over (c_i^{min}, c_i^{max}).

2. (a) Generate a pseudorandom number U uniformly distributed over $(0, 1)$.

   (b) Set $c_e^k = c_e^{min} + (c_e^{max} - c_e^{min})\,U$ to make c_e^k uniformly distributed over (c_e^{min}, c_e^{max}).

3. (a) Generate a pseudorandom number U uniformly distributed over $(0, 1)$.

(b) Set $c_h^k = c_h^{min} + (c_h^{max} - c_h^{min})\,U$ to make c_h^k uniformly distributed over (c_h^{min}, c_h^{max}).

4. (a) Generate a pseudorandom number U uniformly distributed over $(0,1)$.

   (b) Set $c_s^k = c_s^{min} + (c_s^{max} - c_s^{min})\,U$ to make c_s^k uniformly distributed over (c_s^{min}, c_s^{max}).

5. (a) Generate a pseudorandom number U uniformly distributed over $(0,1)$.

   (b) Set $i^k = i^{min} + (i^{max} - i^{min})\,U$ to make i^k uniformly distributed over (i^{min}, i^{max}).

Subprocedure 4.3 is proposed to calculate the *RE-TLB*, which is used to evaluate the performance of Algorithm 4.1.

Subprocedure 4.3. RE-TLB

1. Calculate the cost given by the heuristic algorithm by adding up $Importing\,[k]$, $Exporting\,[k]$, $Holding\,[k]$ and $Stockout\,[k]$ for all $1 \leqslant k \leqslant K$, denoted as $Sum^{Algorithm}$.

2. Calculate the *TLB* by adding up $Importing^{TLB}\,[k]$, $Exporting^{TLB}\,[k]$,

$Holding^{TLB}[k]$ and $Stockout^{TLB}[k]$ for all $1 \leqslant k \leqslant K$, denoted as Sum^{TLB}.

3. Calculate the *RE-TLB* as:

$$RE\text{-}TLB = Sum^{Algorithm}/Sum^{TLB} - 1. \tag{4.19}$$

The following simulation procedures are used to simulate the empty container repositioning process between multi-ports over multi-periods and then calculate the *average* relative error with respect to the tight lower bound (*AVG-RE-TLB*) to evaluate the performance of Algorithm 4.1. We run this simulation for numerous ports, ranging from $Min\text{-}K$ to $Max\text{-}K$. For each number of ports, we run $No. of\ Instances$ problem instances, and in each instance we run this simulation $No. of\ Iterations$ times and then calculate the average values to obtain *AVG-RE-TLB*, making the results more reliable.

Simulation Procedures

For each K within $Min\text{-}K \leqslant K \leqslant Max\text{-}K$:

1. Set $number = 1$.

2. If $number \leqslant No. of\ Instances$, do the following; else stop.

3. Randomly generate c_i^k, c_e^k, c_h^k, c_s^k and initial i^k for all $1 \leqslant k \leqslant K$ by Subprocedure 4.2.

4. Calculate A_n^k, S_n^k and $V_n^k(i)$ for all $1 \leqslant n \leqslant N$, $1 \leqslant k \leqslant K$ and $0 \leqslant i \leqslant M + R$ by Algorithm 3.1.

5. Set $i_1^k = i^k$ to copy initial i^k to i_1^k for all $1 \leqslant k \leqslant K$.

6. Set $Sum^{RE\text{-}TLB} = 0$.

7. Set $Sum\text{-}Importing[k]$, $Sum\text{-}Exporting[k]$, $Sum\text{-}Holding[k]$ and $Sum\text{-}Stockout[k]$; $Sum\text{-}Importing^{TLB}[k]$, $Sum\text{-}Exporting^{TLB}[k]$, $Sum\text{-}Holding^{TLB}[k]$ and $Sum\text{-}Stockout^{TLB}[k]$ equal to 0 for all $1 \leqslant k \leqslant K$.

8. Do this step $No.\,of\;Iterations$ times, in each times of execution:

   (a) Set $Importing[k]$, $Exporting[k]$, $Holding[k]$ and $Stockout[k]$; $Importing^{TLB}[k]$, $Exporting^{TLB}[k]$, $Holding^{TLB}[k]$ and $Stockout^{TLB}[k]$ equal to 0 for all $1 \leqslant k \leqslant K$.

   (b) Reposition the empty containers by Subprocedure 4.1.

   (c) Calculate the corresponding $RE\text{-}TLB$ by Subprocedure 4.3.

   (d) Reset $Sum^{RE\text{-}TLB} = Sum^{RE\text{-}TLB} + RE\text{-}TLB$.

(e) Add the values of $Importing\,[k]$, $Exporting\,[k]$, $Holding\,[k]$ and $Stockout\,[k]$; $Importing^{TLB}\,[k]$, $Exporting^{TLB}\,[k]$, $Holding^{TLB}\,[k]$ and $Stockout^{TLB}\,[k]$ to $Sum\text{-}Importing\,[k]$, $Sum\text{-}Exporting\,[k]$, $Sum\text{-}Holding\,[k]$ and $Sum\text{-}Stockout\,[k]$; $Sum\text{-}Importing^{TLB}\,[k]$, $Sum\text{-}Exporting^{TLB}\,[k]$, $Sum\text{-}Holding^{TLB}\,[k]$ and $Sum\text{-}Stockout^{TLB}\,[k]$, respectively, for all $1 \leqslant k \leqslant K$.

(f) Reset $i^k = i^k_1$ to restore initial i^k for all $1 \leqslant k \leqslant K$.

9. (a) Calculate the average values of $Importing\,[k]$, $Exporting\,[k]$, $Holding\,[k]$ and $Stockout\,[k]$; $Importing^{TLB}\,[k]$, $Exporting^{TLB}\,[k]$, $Holding^{TLB}\,[k]$ and $Stockout^{TLB}\,[k]$ as $Sum\text{-}Importing\,[k]$, $Sum\text{-}Exporting\,[k]$, $Sum\text{-}Holding\,[k]$ and $Sum\text{-}Stockout\,[k]$; $Sum\text{-}Importing^{TLB}\,[k]$, $Sum\text{-}Exporting^{TLB}\,[k]$, $Sum\text{-}Holding^{TLB}\,[k]$ and $Sum\text{-}Stockout^{TLB}\,[k]$ divided by $No.\,of\ Iterations$, respectively, for all $1 \leqslant k \leqslant K$.

   (b) Reset the values of $Importing\,[k]$, $Exporting\,[k]$, $Holding\,[k]$ and $Stockout\,[k]$; $Importing^{TLB}\,[k]$, $Exporting^{TLB}\,[k]$, $Holding^{TLB}\,[k]$ and $Stockout^{TLB}\,[k]$ with their average values, respectively, for all $1 \leqslant k \leqslant K$.

10. Calculate the *AVG-RE-TLB*:

$$AVG\text{-}RE\text{-}TLB = Sum^{RE\text{-}TLB}/No.\,of\ Iterations. \tag{4.20}$$

11. Output the following values:

    (a) c_i^k, c_e^k, c_h^k, c_s^k and initial i^k for all $1 \leqslant k \leqslant K$.

    (b) A_n^k and S_n^k for all $1 \leqslant n \leqslant N$ and $1 \leqslant k \leqslant K$.

    (c) $Importing\,[k]$, $Exporting\,[k]$, $Holding\,[k]$ and $Stockout\,[k]$ for all $1 \leqslant k \leqslant K$.

    (d) $Importing^{TLB}\,[k]$, $Exporting^{TLB}\,[k]$, $Holding^{TLB}\,[k]$ and $Stockout^{TLB}\,[k]$ for all $1 \leqslant k \leqslant K$.

    (e) $AVG\text{-}RE\text{-}TLB$.

12. Reset $number = number + 1$, go to step 2.

Due to the same reason stated in Section 3.3 for the single-port case, we continue to use the normal distribution and uniform distribution to conduct the simulation, and the planning horizon also contains 12 consecutive decision periods. The following parameters are used in the simulation.

$M = 1000,\ \ R = 50,\ \ N = 12,\ \ \alpha = 0.99,\ \ Min\text{-}K = 5,\ \ Max\text{-}K = 50,$

$No.\,of\ Instances = 10,\ \ No.\,of\ Iterations = 100,\ \ (c_i^{min}, c_i^{max}) = (140, 160),$

$(c_e^{min}, c_e^{max}) = (140, 160)$, $(c_h^{min}, c_h^{max}) = (160, 200)$,

$(c_s^{min}, c_s^{max}) = (900, 1100)$, $(i^{min}, i^{max}) = (0, 40)$.

We summarize the results, i.e., *AVG-RE-TLB*, in Tables A.1 and A.2 for the normal distribution and discrete uniform distribution, respectively. We next compute the maximum value, minimum value, average value and standard deviation of the 10 problem instances for each number of ports, abbreviated as Max, Min, Avg and SD in Tables A.1 and A.2 for the normal distribution and discrete uniform distribution, respectively.

The data corresponding to Max, Min and Avg for all the ports are plotted in Figures 4.1 and 4.3 for the normal distribution and discrete uniform distribution, respectively. The data corresponding to SD for all the ports are plotted in Figures 4.2 and 4.4 for the normal distribution and discrete uniform distribution, respectively.

The simulation is conducted by running a C++ program on a PC with Pentium 4 CPU 3GHz and 504MB of RAM.

4.3.2 Case I: Normal Distribution

Suppose random variables Z_1^k and Z_2^k for all k follow the same normal distribution with $\mu_{Z_1^k} = \mu_{Z_2^k}$ and $\sigma^2_{Z_1^k} = \sigma^2_{Z_2^k} = 50$. Since Z_1^k and Z_2^k are independent, random variable $Z^k = Z_1^k - Z_2^k \left(-R \leqslant z^k \leqslant R\right)$ follows the normal distribution with $\mu_{Z^k} = \mu_{Z_1^k} - \mu_{Z_2^k} = 0$ and $\sigma^2_{Z^k} = \sigma^2_{Z_1^k} + \sigma^2_{Z_2^k} = 100$. The results are shown in Figures 4.1 and 4.2.

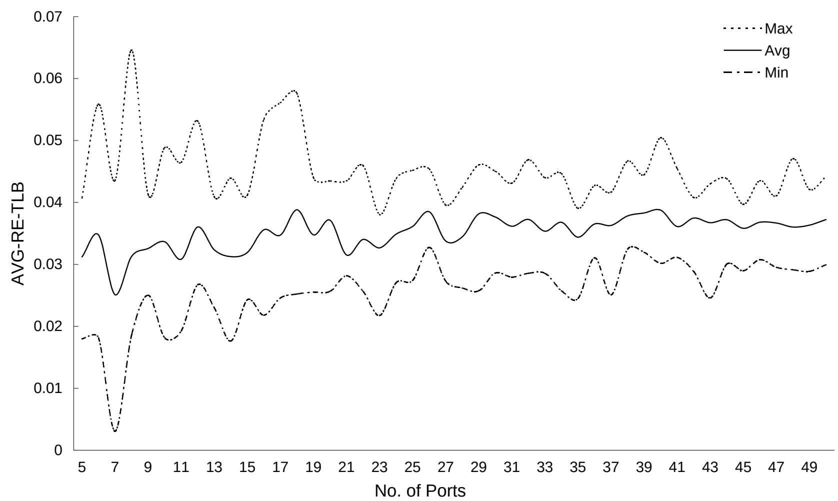

Figure 4.1: *AVG-RE-TLB* under the Normal Distribution

Figure 4.1 shows that the maximum values for all the cases are less than 7 per cent, and the average values appear to be stable at 3 per cent level, which means that Algorithm 4.1 is very effective under the normal distribution.

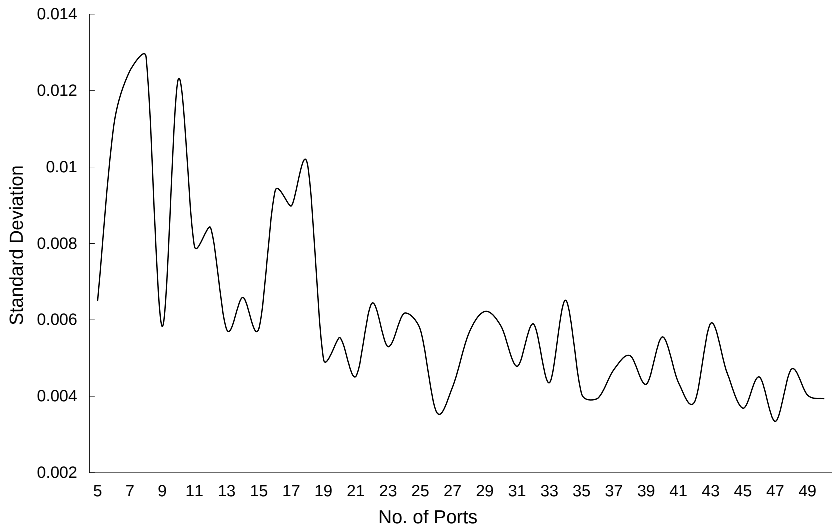

Figure 4.2: Standard Deviation under the Normal Distribution

Figure 4.2 shows that the standard deviation approaches 0 with the number of ports increasing, which is due to the fact that the AVG-RE-TLB approaches a constant as the number of ports increases. Therefore, the range in Figure 4.1 also approaches 0, namely, two dashed lines converge to the middle real line with the number of ports increasing, which means that the stability of Algorithm 4.1 improves as the number of ports increases under the normal distribution.

4.3.3 Case II: Discrete Uniform Distribution

Suppose random variables Z_1^k and Z_2^k for all k follow the same discrete uniform distribution in the interval $[0, R]$. Thus random variable $Z^k = Z_1^k - Z_2^k \left(-R \leqslant z^k \leqslant R\right)$ follows the discrete triangular distribution established in Lemma 3.1. The results are shown in Figures 4.3 and 4.4.

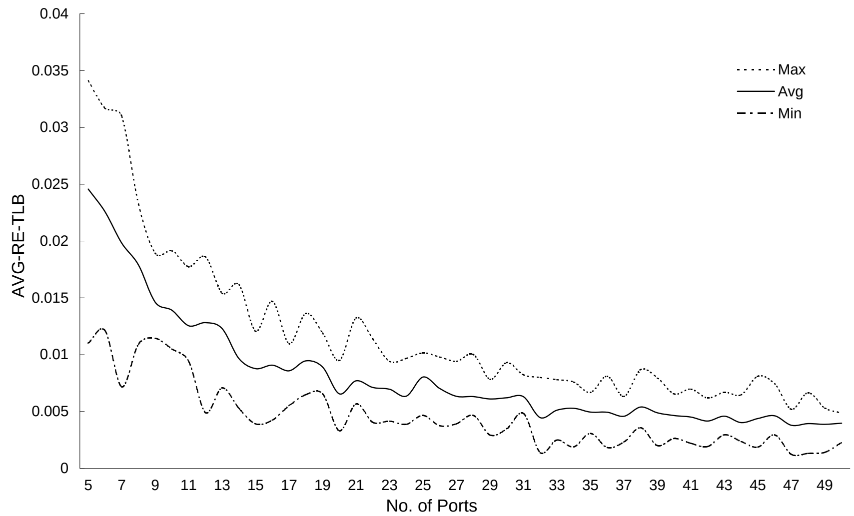

Figure 4.3: *AVG-RE-TLB* under the Discrete Uniform Distribution

Figure 4.3 shows that the maximum values for all the cases are less than 4 per cent and the average values approach 0 as the number of ports increases, which means that Algorithm 4.1 is very effective under the discrete uniform distribution.

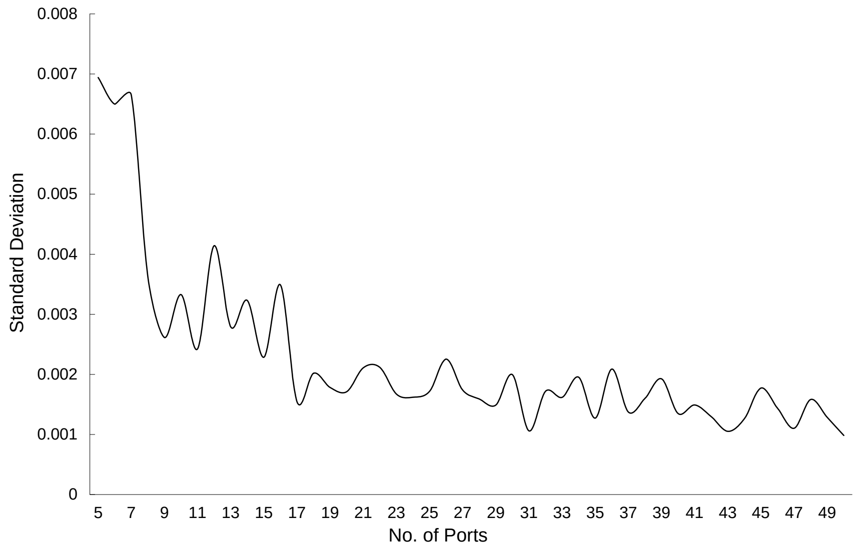

Figure 4.4: Standard Deviation under the Discrete Uniform Distribution

Figure 4.4 shows that the standard deviation approaches 0 with the number of ports increasing, which is due to the fact that the $AVG\text{-}RE\text{-}TLB$ approaches a constant as the number of ports increases. Therefore, the range in Figure 4.3 also approaches 0, namely, two dashed lines converge to the middle real line with the number of ports increasing, which means that the stability of Algorithm 4.1 improves as the number of ports increases under the discrete uniform distribution.

4.4 Summary

We have extended the single-port case to multi-ports in this chapter. We have mathematically formulated the multi-port empty container repositioning problem with stochastic demand and lost sales. After determining a tight lower bound on the cost function, we have introduced the concept of relative error with respect to the tight lower bound, which can be used to measure the performance of Algorithm 4.1 in accordance with Lemma 4.1. Based on the two-threshold optimal policy established for a single port in Theorem 3.1, we have developed a polynomial time Algorithm 4.1 to find an approximate repositioning policy for multi-ports and then use simulation method to test its performance. Simulation results show that the average relative error with respect to the tight lower bound is within 5 per cent under the normal distribution and uniform distribution, respectively. Thus the underlying true relative error must be within 5 per cent or even smaller according to Lemma 4.1, which indicates Algorithm 4.1 performs very effectively for the multi-port empty container repositioning. Furthermore, Algorithm 4.1 performs very efficiently due to its polynomial running time. The stability of Algorithm 4.1 improves as the number of ports increases. More importantly, Algorithm 4.1 is easy to understand and implement from a practical perspective because of its simplicity.

Chapter 5

Concluding Remarks

In this study, we have analyzed the multi-period empty container repositioning problem with stochastic demand and lost sales. Maritime container shipping is a highly competitive industry. Therefore, we assume that unsatisfied customer demand due to the unavailability of empty containers will be lost forever, and will incur a stockout cost, i.e., we assume lost sales scenario in our model. We do not consider leasing policy as an option to supply empty containers in our model based on the reasonable justifications. We aim to establish an effective empty container repositioning policy with the objective to minimize the total operating cost, i.e., container holding cost, stockout cost, importing cost and exporting cost.

First, we have mathematically formulated the single-port case as an inventory problem over a finite horizon with stochastic import and export of empty containers. We have analytically established the two-threshold optimal policy for a single port, i.e., for period n: importing empty containers up to A_n when the number of empty containers in the port is fewer than A_n; exporting empty containers down to S_n when the number of empty containers in the port is more than S_n; and doing nothing, otherwise. We have also developed a polynomial time algorithm to numerically calculate the two thresholds A_n and S_n for each period. We have provided two examples to illustrate the solution procedures based on the normal distribution and uniform distribution, respectively. The results show that the proposed algorithm performs highly effectively and efficiently.

Next, we have extended the single-port case to multi-ports. We have also mathematically formulated the multi-port problem and determined a tight lower bound on the cost function. We then introduce the concept of relative error with respect to the tight lower bound, which is used to measure the performance of the proposed algorithm. Based on the two-threshold optimal policy established for a single port, we have developed a polynomial time algorithm to find an approximate repositioning policy for multi-ports and then use simulation approach to test its performance. The simulation results show that the proposed approximate repositioning algorithm performs very effectively since the calculated average relative error with respect to the tight lower bound is within 5 per cent under the normal distribution and uniform distribution, respectively. Furthermore, the algorithm performs very efficiently as a result of its polynomial running time. The stability of the proposed algorithm improves as the number of ports increases. More importantly, the proposed approximate repositioning algorithm features being easy to understand and implement from a practical perspective. In reality, a liner shipping company manager can first calculate the two thresholds for all ports in all periods at the beginning of the whole planning horizon; then at the beginning of each decision period, the manager can apply the repositioning algorithm to determine the specific repositioning policy for this period.

There are several promising directions for future research. For example, it can examine the effect of shipping capacity on empty container repositioning, i.e., it means that there is an upper bound on the number of repositioning empty containers, and thus the recursive relation is minimized under the constraint, which may change the structure of the optimal policy. Furthermore, this study can be extended by analyzing the empty container repositioning problem over an infinite planning horizon, e.g., one can discuss the convergence of two thresholds in an infinite setting.

Appendix A

Simulated Data

Table A.1: Simulated Data for Multi-ports under the Normal Distribution

No. of Ports	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8	Case 9	Case 10	Max	Min	Avg	SD
5	0.0264	0.0390	0.0358	0.0407	0.0298	0.0300	0.0288	0.0180	0.0322	0.0312	0.0407	0.0180	0.0312	0.0065
6	0.0381	0.0559	0.0333	0.0407	0.0230	0.0365	0.0413	0.0379	0.0181	0.0230	0.0559	0.0181	0.0348	0.0111
7	0.0360	0.0249	0.0436	0.0356	0.0031	0.0190	0.0219	0.0195	0.0354	0.0121	0.0436	0.0031	0.0251	0.0125
8	0.0186	0.0252	0.0337	0.0359	0.0245	0.0646	0.0323	0.0299	0.0222	0.0259	0.0646	0.0186	0.0313	0.0129
9	0.0306	0.0388	0.0277	0.0394	0.0307	0.0250	0.0307	0.0413	0.0356	0.0259	0.0413	0.0250	0.0326	0.0058
10	0.0481	0.0306	0.0182	0.0207	0.0380	0.0203	0.0255	0.0379	0.0488	0.0487	0.0488	0.0182	0.0337	0.0123
11	0.0265	0.0334	0.0193	0.0333	0.0329	0.0374	0.0313	0.0216	0.0465	0.0261	0.0465	0.0193	0.0308	0.0079
12	0.0275	0.0335	0.0531	0.0409	0.0305	0.0267	0.0418	0.0299	0.0335	0.0429	0.0531	0.0267	0.0360	0.0084
13	0.0347	0.0370	0.0231	0.0321	0.0305	0.0230	0.0409	0.0323	0.0369	0.0330	0.0409	0.0230	0.0324	0.0057
14	0.0439	0.0305	0.0325	0.0307	0.0279	0.0358	0.0326	0.0285	0.0324	0.0176	0.0439	0.0176	0.0313	0.0066
15	0.0333	0.0273	0.0243	0.0382	0.0308	0.0412	0.0384	0.0254	0.0295	0.0307	0.0412	0.0243	0.0319	0.0058
16	0.0455	0.0357	0.0340	0.0218	0.0272	0.0327	0.0298	0.0324	0.0434	0.0534	0.0534	0.0218	0.0356	0.0094
17	0.0333	0.0561	0.0307	0.0392	0.0390	0.0343	0.0299	0.0347	0.0246	0.0255	0.0561	0.0246	0.0347	0.0090
18	0.0366	0.0252	0.0576	0.0382	0.0340	0.0542	0.0300	0.0397	0.0394	0.0332	0.0576	0.0252	0.0388	0.0101
19	0.0327	0.0345	0.0440	0.0327	0.0325	0.0383	0.0353	0.0394	0.0255	0.0328	0.0440	0.0255	0.0348	0.0050
20	0.0326	0.0257	0.0379	0.0333	0.0407	0.0388	0.0420	0.0348	0.0420	0.0435	0.0435	0.0257	0.0371	0.0055

Note: Table A.1 continues on the next page.

No. of Ports	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8	Case 9	Case 10	Max	Min	Avg	SD
21	0.0297	0.0329	0.0283	0.0282	0.0321	0.0318	0.0435	0.0308	0.0284	0.0296	0.0435	0.0282	0.0315	0.0045
22	0.0376	0.0409	0.0256	0.0314	0.0459	0.0292	0.0381	0.0292	0.0285	0.0339	0.0459	0.0256	0.0340	0.0064
23	0.0217	0.0381	0.0316	0.0315	0.0341	0.0372	0.0361	0.0256	0.0343	0.0366	0.0381	0.0217	0.0327	0.0053
24	0.0294	0.0270	0.0407	0.0358	0.0277	0.0376	0.0410	0.0439	0.0370	0.0288	0.0439	0.0270	0.0349	0.0062
25	0.0427	0.0286	0.0355	0.0349	0.0274	0.0331	0.0381	0.0349	0.0409	0.0452	0.0452	0.0274	0.0361	0.0057
26	0.0454	0.0365	0.0404	0.0368	0.0384	0.0328	0.0384	0.0355	0.0385	0.0424	0.0454	0.0328	0.0385	0.0036
27	0.0368	0.0335	0.0373	0.0380	0.0284	0.0303	0.0396	0.0273	0.0315	0.0350	0.0396	0.0273	0.0338	0.0042
28	0.0322	0.0298	0.0353	0.0273	0.0389	0.0350	0.0262	0.0357	0.0424	0.0418	0.0424	0.0262	0.0344	0.0056
29	0.0461	0.0336	0.0418	0.0257	0.0318	0.0396	0.0399	0.0433	0.0368	0.0431	0.0461	0.0257	0.0382	0.0062
30	0.0286	0.0332	0.0339	0.0436	0.0450	0.0446	0.0313	0.0412	0.0378	0.0372	0.0450	0.0286	0.0376	0.0058
31	0.0417	0.0291	0.0349	0.0372	0.0387	0.0357	0.0364	0.0280	0.0431	0.0368	0.0431	0.0280	0.0362	0.0048
32	0.0453	0.0401	0.0324	0.0367	0.0469	0.0399	0.0286	0.0348	0.0316	0.0364	0.0469	0.0286	0.0373	0.0059
33	0.0365	0.0440	0.0342	0.0324	0.0319	0.0382	0.0350	0.0335	0.0393	0.0286	0.0440	0.0286	0.0354	0.0044
34	0.0416	0.0435	0.0404	0.0279	0.0258	0.0308	0.0376	0.0386	0.0372	0.0446	0.0446	0.0258	0.0368	0.0065
35	0.0336	0.0331	0.0245	0.0391	0.0368	0.0334	0.0352	0.0387	0.0350	0.0345	0.0391	0.0245	0.0344	0.0041
36	0.0391	0.0358	0.0361	0.0416	0.0367	0.0328	0.0376	0.0427	0.0317	0.0311	0.0427	0.0311	0.0365	0.0040

Note: Table A.1 continues on the next page.

No. of Ports	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8	Case 9	Case 10	Max	Min	Avg	SD
37	0.0341	0.0355	0.0251	0.0371	0.0368	0.0417	0.0409	0.0353	0.0363	0.0402	0.0417	0.0251	0.0363	0.0047
38	0.0396	0.0356	0.0356	0.0467	0.0438	0.0328	0.0326	0.0325	0.0423	0.0368	0.0467	0.0325	0.0378	0.0051
39	0.0411	0.0445	0.0396	0.0332	0.0411	0.0421	0.0386	0.0320	0.0383	0.0327	0.0445	0.0320	0.0383	0.0043
40	0.0302	0.0376	0.0338	0.0438	0.0504	0.0413	0.0389	0.0362	0.0379	0.0373	0.0504	0.0302	0.0388	0.0056
41	0.0327	0.0347	0.0364	0.0413	0.0371	0.0311	0.0454	0.0327	0.0360	0.0337	0.0454	0.0311	0.0361	0.0044
42	0.0327	0.0405	0.0395	0.0380	0.0382	0.0391	0.0367	0.0288	0.0408	0.0404	0.0408	0.0288	0.0375	0.0039
43	0.0304	0.0418	0.0412	0.0324	0.0376	0.0246	0.0410	0.0430	0.0364	0.0390	0.0430	0.0246	0.0367	0.0059
44	0.0377	0.0325	0.0324	0.0365	0.0379	0.0370	0.0407	0.0439	0.0434	0.0300	0.0439	0.0300	0.0372	0.0046
45	0.0370	0.0390	0.0290	0.0337	0.0345	0.0397	0.0390	0.0385	0.0309	0.0369	0.0397	0.0290	0.0358	0.0037
46	0.0365	0.0336	0.0378	0.0429	0.0406	0.0308	0.0372	0.0435	0.0314	0.0339	0.0435	0.0308	0.0368	0.0045
47	0.0373	0.0410	0.0345	0.0378	0.0411	0.0381	0.0360	0.0295	0.0352	0.0362	0.0411	0.0295	0.0367	0.0033
48	0.0471	0.0384	0.0364	0.0378	0.0342	0.0337	0.0326	0.0357	0.0291	0.0353	0.0471	0.0291	0.0360	0.0047
49	0.0322	0.0289	0.0391	0.0421	0.0383	0.0330	0.0361	0.0384	0.0399	0.0354	0.0421	0.0289	0.0363	0.0040
50	0.0404	0.0383	0.0374	0.0338	0.0385	0.0300	0.0366	0.0341	0.0442	0.0391	0.0442	0.0300	0.0372	0.0039

Note: Case n in this table represents the n-th problem instance.

Table A.2: Simulated Data for Multi-ports under the Discrete Uniform Distribution

No. of Ports	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8	Case 9	Case 10	Max	Min	Avg	SD
5	0.0159	0.0341	0.0224	0.0272	0.0110	0.0302	0.0252	0.0258	0.0232	0.0308	0.0341	0.0110	0.0246	0.0069
6	0.0245	0.0121	0.0189	0.0277	0.0317	0.0228	0.0146	0.0215	0.0313	0.0208	0.0317	0.0121	0.0226	0.0065
7	0.0225	0.0113	0.0253	0.0202	0.0193	0.0072	0.0310	0.0197	0.0202	0.0216	0.0310	0.0072	0.0198	0.0067
8	0.0164	0.0193	0.0204	0.0232	0.0172	0.0162	0.0109	0.0163	0.0165	0.0226	0.0232	0.0109	0.0179	0.0036
9	0.0133	0.0148	0.0172	0.0177	0.0189	0.0125	0.0154	0.0126	0.0114	0.0123	0.0189	0.0114	0.0146	0.0026
10	0.0108	0.0191	0.0190	0.0105	0.0159	0.0158	0.0122	0.0110	0.0116	0.0131	0.0191	0.0105	0.0139	0.0033
11	0.0135	0.0120	0.0094	0.0124	0.0129	0.0126	0.0094	0.0142	0.0111	0.0178	0.0178	0.0094	0.0125	0.0024
12	0.0121	0.0105	0.0186	0.0136	0.0083	0.0133	0.0049	0.0134	0.0162	0.0172	0.0186	0.0049	0.0128	0.0041
13	0.0071	0.0145	0.0154	0.0130	0.0090	0.0127	0.0124	0.0135	0.0099	0.0154	0.0154	0.0071	0.0123	0.0028
14	0.0114	0.0127	0.0162	0.0053	0.0097	0.0094	0.0077	0.0097	0.0057	0.0089	0.0162	0.0053	0.0097	0.0032
15	0.0104	0.0121	0.0039	0.0095	0.0099	0.0072	0.0088	0.0105	0.0074	0.0080	0.0121	0.0039	0.0088	0.0023
16	0.0147	0.0055	0.0126	0.0109	0.0042	0.0066	0.0103	0.0052	0.0106	0.0100	0.0147	0.0042	0.0091	0.0035
17	0.0067	0.0055	0.0086	0.0092	0.0084	0.0091	0.0097	0.0110	0.0095	0.0081	0.0110	0.0055	0.0086	0.0015
18	0.0105	0.0075	0.0083	0.0101	0.0107	0.0065	0.0091	0.0136	0.0101	0.0081	0.0136	0.0065	0.0095	0.0020
19	0.0104	0.0083	0.0068	0.0069	0.0096	0.0105	0.0087	0.0119	0.0093	0.0066	0.0119	0.0066	0.0089	0.0018
20	0.0082	0.0058	0.0095	0.0065	0.0060	0.0077	0.0033	0.0054	0.0073	0.0058	0.0095	0.0033	0.0066	0.0017

Note: Table A.2 continues on the next page.

No. of Ports	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8	Case 9	Case 10	Max	Min	Avg	SD
21	0.0083	0.0065	0.0071	0.0071	0.0078	0.0132	0.0060	0.0079	0.0075	0.0057	0.0132	0.0057	0.0077	0.0021
22	0.0057	0.0073	0.0078	0.0114	0.0079	0.0043	0.0072	0.0085	0.0041	0.0070	0.0114	0.0041	0.0071	0.0021
23	0.0072	0.0070	0.0080	0.0094	0.0086	0.0042	0.0074	0.0073	0.0046	0.0058	0.0094	0.0042	0.0070	0.0017
24	0.0068	0.0071	0.0097	0.0064	0.0064	0.0041	0.0039	0.0068	0.0057	0.0067	0.0097	0.0039	0.0063	0.0016
25	0.0047	0.0101	0.0063	0.0069	0.0089	0.0089	0.0102	0.0078	0.0089	0.0078	0.0102	0.0047	0.0080	0.0017
26	0.0076	0.0062	0.0054	0.0091	0.0038	0.0094	0.0091	0.0056	0.0098	0.0043	0.0098	0.0038	0.0070	0.0023
27	0.0048	0.0079	0.0070	0.0042	0.0070	0.0094	0.0068	0.0054	0.0039	0.0069	0.0094	0.0039	0.0063	0.0017
28	0.0073	0.0050	0.0047	0.0100	0.0063	0.0047	0.0061	0.0072	0.0059	0.0060	0.0100	0.0047	0.0063	0.0016
29	0.0078	0.0029	0.0061	0.0078	0.0050	0.0066	0.0073	0.0054	0.0068	0.0055	0.0078	0.0029	0.0061	0.0015
30	0.0060	0.0088	0.0059	0.0035	0.0035	0.0079	0.0055	0.0052	0.0093	0.0066	0.0093	0.0035	0.0062	0.0020
31	0.0052	0.0069	0.0058	0.0070	0.0064	0.0073	0.0082	0.0055	0.0057	0.0048	0.0082	0.0048	0.0063	0.0011
32	0.0034	0.0014	0.0044	0.0038	0.0034	0.0045	0.0051	0.0055	0.0052	0.0080	0.0080	0.0014	0.0045	0.0017
33	0.0043	0.0075	0.0025	0.0039	0.0048	0.0049	0.0078	0.0059	0.0054	0.0041	0.0078	0.0025	0.0051	0.0016
34	0.0019	0.0076	0.0070	0.0021	0.0068	0.0056	0.0051	0.0055	0.0046	0.0065	0.0076	0.0019	0.0053	0.0020
35	0.0067	0.0053	0.0040	0.0044	0.0032	0.0031	0.0049	0.0055	0.0062	0.0063	0.0067	0.0031	0.0049	0.0013
36	0.0068	0.0018	0.0069	0.0053	0.0040	0.0081	0.0031	0.0066	0.0030	0.0036	0.0081	0.0018	0.0049	0.0021

Note: Table A.2 continues on the next page.

No. of Ports	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7	Case 8	Case 9	Case 10	Max	Min	Avg	SD
37	0.0031	0.0053	0.0056	0.0023	0.0050	0.0057	0.0042	0.0063	0.0054	0.0029	0.0063	0.0023	0.0046	0.0014
38	0.0075	0.0049	0.0044	0.0087	0.0049	0.0062	0.0039	0.0051	0.0036	0.0048	0.0087	0.0036	0.0054	0.0016
39	0.0020	0.0056	0.0033	0.0042	0.0042	0.0075	0.0065	0.0080	0.0039	0.0038	0.0080	0.0020	0.0049	0.0019
40	0.0060	0.0059	0.0026	0.0066	0.0050	0.0050	0.0046	0.0047	0.0031	0.0030	0.0066	0.0026	0.0046	0.0013
41	0.0055	0.0022	0.0066	0.0035	0.0040	0.0070	0.0034	0.0044	0.0037	0.0047	0.0070	0.0022	0.0045	0.0015
42	0.0033	0.0038	0.0019	0.0062	0.0036	0.0034	0.0048	0.0056	0.0054	0.0037	0.0062	0.0019	0.0042	0.0013
43	0.0046	0.0049	0.0041	0.0044	0.0056	0.0037	0.0039	0.0067	0.0051	0.0030	0.0067	0.0030	0.0046	0.0011
44	0.0036	0.0032	0.0045	0.0024	0.0026	0.0050	0.0065	0.0050	0.0033	0.0043	0.0065	0.0024	0.0040	0.0013
45	0.0056	0.0041	0.0046	0.0039	0.0036	0.0028	0.0019	0.0058	0.0034	0.0081	0.0081	0.0019	0.0044	0.0018
46	0.0043	0.0029	0.0042	0.0047	0.0034	0.0041	0.0074	0.0032	0.0054	0.0064	0.0074	0.0029	0.0046	0.0014
47	0.0029	0.0042	0.0045	0.0033	0.0040	0.0052	0.0045	0.0012	0.0038	0.0043	0.0052	0.0012	0.0038	0.0011
48	0.0032	0.0044	0.0028	0.0067	0.0038	0.0049	0.0013	0.0023	0.0055	0.0044	0.0067	0.0013	0.0039	0.0016
49	0.0040	0.0021	0.0040	0.0040	0.0052	0.0014	0.0036	0.0052	0.0039	0.0053	0.0053	0.0014	0.0039	0.0013
50	0.0048	0.0024	0.0046	0.0047	0.0049	0.0023	0.0041	0.0048	0.0037	0.0035	0.0049	0.0023	0.0040	0.0010

Note: Case n in this table represents the n-th problem instance.

Appendix B

Papers Arising from the Thesis

- Zhang, B., C.T. Ng, T.C.E. Cheng. A stochastic dynamic model for empty container management in a single port. Under 2nd round review in *Journal of the Operational Research Society*.
- Zhang, B., C.T. Ng. A threshold control based heuristic algorithm for empty container repositioning between multi-ports with stochastic demand and lost sales. *Working Paper*.

Appendix C

Conference Presentations

- Zhang, B., C.T. Ng, T.C.E. Cheng. Empty container management with lost sales in a single port. *The Second POMS-HK International Conference*, Hong Kong, January 6-7, 2011.
- Zhang, B., C.T. Ng. Empty container allocation between multi-ports with lost sales. *The 22nd Annual POM Conference*, Reno, Nevada, USA, April 29 - May 2, 2011.

References

Bandeira, D.L., J.L. Becker, D. Borenstein. 2009. A DSS for integrated distribution of empty and full containers. *Decision Support Systems* **47** 383–397.

Cheang, B., A. Lim. 2005. A network flow based method for the distribution of empty containers. *International Journal of Computer Applications in Technology* **22** 198–204.

Cheung, R.K., C.Y. Chen. 1998. A two-stage stochastic network model and solution methods for the dynamic empty container allocation problem. *Transportation Science* **32** 142–162.

Choong, S.T., M.H. Cole, E. Kutanoglu. 2002. Empty container management for intermodal transportation networks. *Transportation Research Part E* **38** 423–438.

Clarkson. 2011a. *Liner register*. Clarkson Research Studies Ltd.

Clarkson. 2011b. *Shipping review and outlook*. Clarkson Research Studies Ltd.

Crainic, T.G., M. Gendreau, P. Dejax. 1993. Dynamic and stochastic models for the allocation of empty containers. *Operations Research* **41** 102–126.

Crainic, T.G., G. Laporte. 1997. Planning models for freight transportation. *European Journal of Operational Research* **97** 409–438.

Dejax, P.J., T.G. Grainic. 1987. A review of empty flows and fleet management models in freight transportation. *Transportation Science* **21** 227–248.

Dong, J.X., D.P. Song. 2009. Container fleet sizing and empty repositioning in liner shipping systems. *Transportation Research Part E* **45** 860–877.

Drewry. 2005. *The Drewry annual container market review and forecast 2005/06*. Drewry Shipping Consultants Ltd.

Du, Y.F., R. Hall. 1997. Fleet sizing and empty equipment redistribution for center-terminal transportation networks. *Management Science* **43** 145–157.

Feng, C.M., C.H. Chang. 2008. Empty container reposition planning for intra-Asia liner shipping. *Maritime Policy and Management* **35** 469–489.

Francesco, M.D., T.G. Crainic, P. Zuddas. 2009. The effect of multi-scenario policies on empty container repositioning. *Transportation Research Part E* **45** 758–770.

Lai, K.K., K. Lam, W.K. Chan. 1995. Shipping container logistics and allocation. *Journal of the Operational Research Society* **46** 687–697.

Lam, S.W., L.H. Lee, L.C. Tang. 2007. An approximate dynamic programming approach for the empty container allocation problem. *Transportation Research Part C* **15** 265–277.

Li, J.A., S.C.H. Leung, Y. Wu, K. Liu. 2007. Allocation of empty containers between multi-ports. *European Journal of Operational Research* **182** 400–412.

Li, J.A., K. Liu, S. Heung, K.K. Lai. 2004. Empty container management in a port with long-run average criterion. *Mathematical and Computer Modelling* **40** 85–100.

Lopez, E. 2003. How do ocean carriers organize the empty containers reposition activity in the USA? *Maritime Policy and Management* **30** 339–355.

Lun, Y.H.V., K.H. Lai, T.C.E. Cheng. 2006. *Shipping and transport logistics*. McGraw-Hill Education (Asia), Singapore.

Olivo, A., P. Zuddas, M.D. Francesco, A. Manca. 2005. An operational model for empty container management. *Maritime Economics and Logistics* **7** 199–222.

Ross, S. 2006. *Simulation*. 4th edition. Elsevier Academic, Burlington.

Shen, W.S., C.M. Khoong. 1995. A DSS for empty container distribution planning. *Decision Support Systems* **15** 75–82.

Shintani, K., A. Imai, E. Nishimura, S. Papadimitriou. 2007. The container shipping network design problem with empty container repositioning. *Transportation Research Part E* **43** 39–59.

Song, D.P. 2005. Optimal threshold control of empty vehicle redistribution in two depot service systems. *IEEE Transactions on Automatic Control* **50** 87–90.

Song, D.P. 2007. Characterizing optimal empty container reposition policy in periodic-review shuttle service systems. *Journal of the Operational Research Society* **58** 122–133.

Song, D.P., J. Carter. 2008. Optimal empty vehicle redistribution for hub-and-spoke transportation systems. *Naval Research Logistics* **55** 156–171.

Song, D.P., J. Carter. 2009. Empty container repositioning in liner shipping. *Maritime Policy and Management* **36** 291–307.

Song, D.P., J.X. Dong. 2008. Empty container management in cyclic shipping routes. *Maritime Economics and Logistics* **10** 335–361.

Song, D.P., J.X. Dong. 2011a. Effectiveness of an empty container repositioning policy with flexible destination ports. *Transport Policy* **18** 92–101.

Song, D.P., J.X. Dong. 2011b. Flow balancing-based empty container repositioning in typical shipping service routes. *Maritime Economics and Logistics* **13** 61–77.

Song, D.P., J.X. Dong, M. Roe. 2010. Optimal container dispatching policy and its structure in a shuttle service with finite capacity and random demands. *International Journal of Shipping and Transport Logistics* **2** 44–58.

Song, D.P., C.F. Earl. 2008. Optimal empty vehicle repositioning and fleet-sizing for two-depot services systems. *European Journal of Operational Research* **185** 760–777.

Song, D.P., J. Zhang, J. Carter, T. Field, J. Marshall, J. Polak, K. Schumacher,

P. Sinha-Ray, J. Woods. 2005. On cost-efficiency of the global container shipping network. *Maritime Policy and Management* **32** 15–30.

Song, D.P., Q. Zhang. 2010. A fluid flow model for empty container repositioning policy with a single port and stochastic demand. *SIAM Journal on Control and Optimization* **48** 3623–3642.

Theofanis, S., M. Boile. 2009. Empty marine container logistics: facts, issues and management strategies. *GeoJournal* **74** 51–65.

UNCTAD. 2005. *Review of maritime transport*. United Nations Publication.

UNCTAD. 2006. *Review of maritime transport*. United Nations Publication.

UNCTAD. 2007. *Review of maritime transport*. United Nations Publication.

UNCTAD. 2008. *Review of maritime transport*. United Nations Publication.

UNCTAD. 2009. *Review of maritime transport*. United Nations Publication.

UNCTAD. 2010. *Review of maritime transport*. United Nations Publication.