Efficient System and Algorithms For Container Storage

Chen Chuanyu

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirement for the degree of
Master of Engineering

2005

Abstract

Due to the ever-growing transport demand and the increasing regional competition, container port operators nowadays are facing great challenges to stay competitive and relevant to their customers. Effective and efficient management of container terminal operations has therefore become one of their main concerns and not surprisingly attracts tremendous research interests. The planning of operations at storage yard, where inbound and outbound container flows are handled simultaneously, determines the port efficiency to a great extent.

This work focuses on enhancing the storage capacity of the yard and its responsiveness to the loading and discharging requests. An innovative solution is proposed that stores containers in Split Platform Automated Storage and Retrieval Systems (SP-AS/RSs) to provide independent access to individual storage positions. The capability of random access removes the unproductive yet common reshuffle operations in the conventional stacking yard, and hence boosts the productivity of the yard. The configuration of the proposed SP-AS/RS is optimized using mathematical analysis of platform travel time models and simulations, and two control rules are tested in various application scenarios to support the AS/RS operations. Deployment algorithms are also considered to schedule yard cranes that pick up and deposit containers from/to vehicles, when yard cranes need to be shared among a number of container storage blocks. Simulation results demonstrate the advantages of the proposed system and its promising application in container storage.

Acknowledgements

I would like to express my sincere appreciation to Dr. Huang Shell Ying, my supervisor, for her guidance and constant support during this research. I am also grateful to Dr. Hsu Wen-Jing for his many suggestions and friendly encouragements.

Dr. Hu Yahong supplied me with the preprints of some of her work on AS/RS that gave me a better perspective on my own results. I had the pleasure of meeting the staff of Maritime and Port Authority who shared with me their expertise in port management that makes the study more practical.

Of course, I am grateful to my parents for their patience, love, and great understanding of my absence during this critical period when my father has yet to recover after suffering stroke. I would extend my gratitude to my friends and colleagues for their demonstration of what trust and relationship mean.

Finally, this work is also for Wenli, who consistently brings me joys and laughters.

Chen Chuanyu
December, 2004

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Since the inception of containerization in 1956, intercontinental container ship has been the massive transportation way for decades. More than 90 percent of international cargo moves through major sea ports everyday, and 80 percent of them are in containers [YC99]. Port operators nowadays are facing great challenges posed by the growing transport and subsequently, the increase of vessel sizes. The new generation of ships, the mega container vessels, with a capacity of more than 10,000 TEUs (Twenty-foot Equivalent Unit), are to be in service in the near future. Their dwell time in port, or turn-around time as quoted by most port operators, is set at 24 hours, for which the container ports are expected to achieve an average service rate of 500 TEUs per hour. Compared to the current maximum rate at 280 TEUs per hour, port facilities have to be drastically up-scaled and optimally utilized.

On the other hand, increased regional competition has put further pressure on port operators to stay competitive and relevant to their customers. Lower handling charge, shorter transit time, higher level of service, and wider connectivity to the rest of the world have been identified as the main goals to acquire international competitiveness.

It is therefore not surprising that a number of studies have been conducted in various aspects for optimization of the container terminal operations. As a result, various control policies for unloading, storing, and loading containers have been proposed based on simulation and mathematical formulations. One of the most prominent results is the application of object-oriented approach, in which terminal resources and entities are modelled as individual objects and solutions to the performance problem are found via operation research techniques [GRM98, YC99, IMP+, BGR99]. However, as far as the throughput of container terminals is concerned, the service rate to meet the demand by mega vessels has yet to be achieved in the studies that concentrated on conventional storage yard, where containers are stacked on the ground, side by side and one on top of another.

The main disadvantage of the conventional stacking scheme is that the reshuffling operations, which incur additional unproductive moves, have to be performed in order to retrieve a container from a lower tier. The vehicle requesting the container will have to wait extra time which may cause delays in feeding to the quay cranes. The result could be as serious as lengthening the vessel's turnaround time and consequently a downgrade of service level. Therefore, in order to reduce the chances of retrieving containers that are not on top of the stacks, and also due to the weight constraint of containers, the stacking height is usually restricted to not more than eight (in practice even lower than eight). However, this practice implies a limited utilization of ground space that is scarce and previous.

In response to these observations, this work focuses on the efficient and effective use of storage yard to satisfy the demand of 500 TEUs per hour service rate in future ports. An innovative solution is proposed to store containers in Automated Storage and Retrieval Systems (AS/RS) that provide random access to individual storage positions to enhance the independence of operations on containers, and subsequently achieve significant throughput improvement in hub container terminals.

AS/RS is a computer-controlled automated material handling system that consists of storage racks, input/output (I/O) stations, and a storage/retrieval (S/R) mechanism to transfer loads between I/O stations and individual storage locations. It has been applied successfully in modern warehouse and manufacturing industry because of its advantages in short processing time and high floor-space utilization [SB95, Par01b]. When dealing with AS/RS it is a common practice to search for the optimal *design* of the system and the corresponding optimal *scheduling* for the S/R machine, both of which eventually decide the system performance [HSG76]. There are two issues in system scheduling: allocating storage locations to incoming loads (*Storage Allocation*) and sequencing storage and retrieval requests (*Interleaving*). On storage allocation, several alternative policies: *randomized, full turnover-based,* and *class-based* were proposed by Hausman, *et al.* in [HSG76], and have been discussed in many studies [Boz84, RE89, ER94, Man97], in which travel time models were developed for evaluation of the policies. Researches on sequencing of requests concentrated on the trade-off between efficient travel of the S/R machine and the response time performance, and were mainly based on simulation due to the complicated and dynamic nature of the problem [GHS77, Ber00].

In this study, we follow the approach to establish rules for scheduling operations corresponding to the system design that takes into account the capacity and throughput demand. Additional considerations on the size and weight of the load are taken care of by proposing a new S/R mechanism for the system. The scheduling of cranes to interface with container carriers at a compatible service rate is also investigated in the study. Several algorithms are proposed to achieve practical and efficient crane deployment.

The remainder of this thesis is organized as follows. The subsequent chapter provides a review and discussion on the existing studies of container terminal operations on the conventional storage yard, and as a result, the necessity to obtain a more efficient storage scheme is outlined. In chapter three, previous studies on AS/RS are scrutinized, the applicability of those policies are also discussed. Chapter four presents the design and control of the AS/RS blocks. The configuration of a AS/RS rack is optimized with throughput as criterion using mathematical analysis, the control rules for the system are examined by simulation models. Several examples are used to demonstrate the feasibility and advantage of the proposed schemes. Chapter five addresses the issue of yard crane deployment for the AS/RS yard. Finally, concluding remarks and future directions of the study are given in chapter six.

Chapter 2

Decision-makings in Container Yard Operations

Management of container terminal operations is essentially the allocation and scheduling of the expensive resources such as berths, quay cranes, storage space, yard cranes, and container carriers. Each type of these resources plays an indispensable role in the interlocking processes in a container terminal. We focus our attention on yard management. In this chapter we give a comprehensive review on various decision problems that arise in the planning of container yard operations, and the solutions given in the literature. Some of the techniques introduced herein will be extended to evaluate the AS/RS storage scheme.

2.1 Introduction and background

Container terminals play a fundamental role in intercontinental cargo transportation by serving as an intermodal interface between the sea and the land carriers. Typically, they receive cargos in containers from various transportation devices like vessels or trucks, store them temporarily to account for the differences in arrival times of the sea and the land transport, and transfer them to other transportation devices to be delivered to their destinations. Figure 2.1 is a schematic diagram showing the core operations in a container terminal. These operations, despite the diversity in terminology, are normally grouped into the following sets of processes that occur simultaneously and interactively:

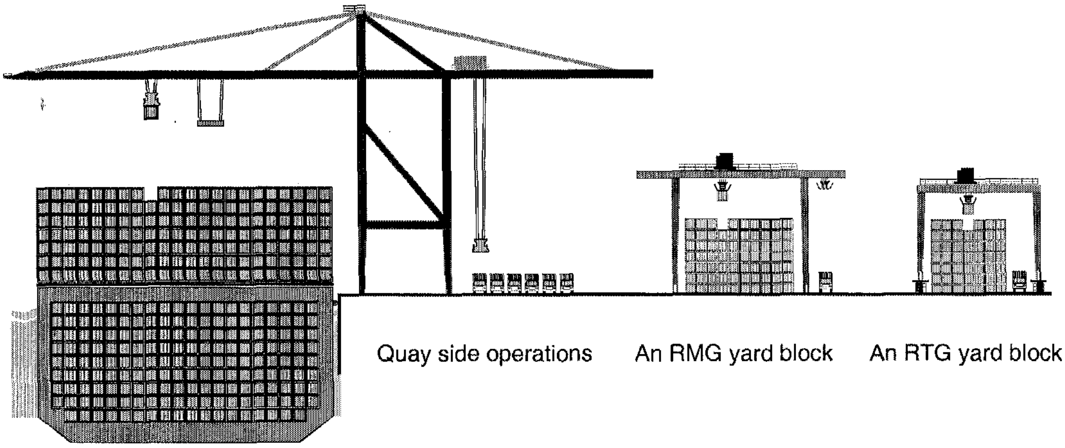

Figure 2.1: An overview of container terminal port operations

(a) Quay side operations: this includes unloading inbound (e.g., import) containers from and loading outbound (e.g., export) containers to the vessels via a number of quay cranes (QC), interacting with the transport system in the terminal. The scheduling of vessel berthing and the allocation of quay cranes are important factors in determining the ship turnaround time.

(b) Transfer operations: this focuses on transferring containers between the berth area and the storage yard by container carriers, such as prime movers, yard trucks, or AGVs (Automated Guided Vehicles). A transport system with efficient rules for dispatching, scheduling, and routing the vehicles would effectively reduce congestions and ensure continuous feeding of the quay cranes.

(c) Yard side operations: this involves the yard space allocation to containers, the sequencing of stacking and un-stacking, and the allocation of equipment, i.e., yard cranes that store and retrieve containers within yard blocks. Current practice reveals that activities taking place in the storage yard tend to be the bottleneck of port operations due to the limited available resources.

These operations combine to generate the work flows in a container terminal, typically starting from the quay side and triggered by the vessel arrival event. The inbound containers are discharged from the vessel and transferred to the designated yard place for pickup by customer (*local import*) or loading to the connecting vessel (*transhipment*). Following that, the outbound containers (*local export* and *transhipment*) are delivered to the quay side and loaded onto the vessel. The vessel departs after the last container is loaded.

There are various performance indices of a container terminal from different standpoints, but ultimately the terminal performance is measured by its service level to the customers, e.g., shipping lines. In alignment, port operators are looking at two typical objectives [CLW⁺03]:

- to minimize the (average)vessel turnaround time, which is a measure of the service level of a terminal to its customers, i.e. the shipping companies.
- to maximize the (average) throughput of the terminal (reflected by the QC rate), which is a measure of the productivity of a terminal.

In order to achieve satisfactory performance, numerous decisions have to be made to manage the daily operations. The complexity lies in the fact that the decisions have an impact on each other. For example, the decisions on the storage locations of containers would directly affect the allocation of yard cranes, the dispatching of the prime movers, and the degree of traffic congestion, and indirectly affect the scheduling of QCs. Moreover, the multiple involvement of different action parties, e.g., berth planning department and yard planning department, and the great difference in the length of their planning horizons further complicate the operation management problem [HVJD99]. Therefore, it becomes evident that it is, if not impossible, very difficult to achieve optimal decisions that favor the above objectives. Logically, hierarchical decomposition of the original problem into simpler sub-problems is an applicable and effective approach to adopt [Psa98].

These sub-problems (or processes) have been commonly identified as follows with different target time domains [YC99]:

- *Berth allocation* that determines the berthing sequence of arriving ships and in turn controls the loading and unloading of containers onboard the ships. It is a multi-objective problem taking into account the vessel turnaround time and space utilization. Berth allocation is performed a few days before a ship arrives.

- *Stowage planning* that assigns to each bay position a particular outbound container with a type matching the preliminary type-based stowage plan provided by shippers. It is normally done by assigning outbound containers in inverse order of ports to be visited by the ships and changes may be required from the shipping companies. Stowage plans are prepared a few hours in advance.

- *Yard planning* that allocates proper storage locations for the inbound containers, the purpose of which is to integrate all activities within the terminal area into a seamless whole. Though not conclusive, the opinion from most port operators is that it is yard planning that is the key to efficient terminal operations. Yard planning is done on a time horizon of weeks.

- *Logistic planning* that coordinates the allocation of resources for handling containers, such as prime movers, and therefore is also called *Resource allocation* in some literatures. Normally logistic plans are decided days beforehand.

Figure 2.2 illustrates them in a typical hierarchical structure. As the combined effect of these decision making tasks, the loading and unloading lists of containers for individual vessels are generated with adjustment according to the situation at hand. Once these lists are determined, the productivity of a container terminal becomes assessable, reflecting the level of optimization of the operation policies.

Among these processes, the planning of yard operations plays an important role for efficient port management due to the limited space and high throughput demand in many terminals. A well designed and planned storage yard would largely improve port performance with high space utilization. In the following sections we will review various solutions

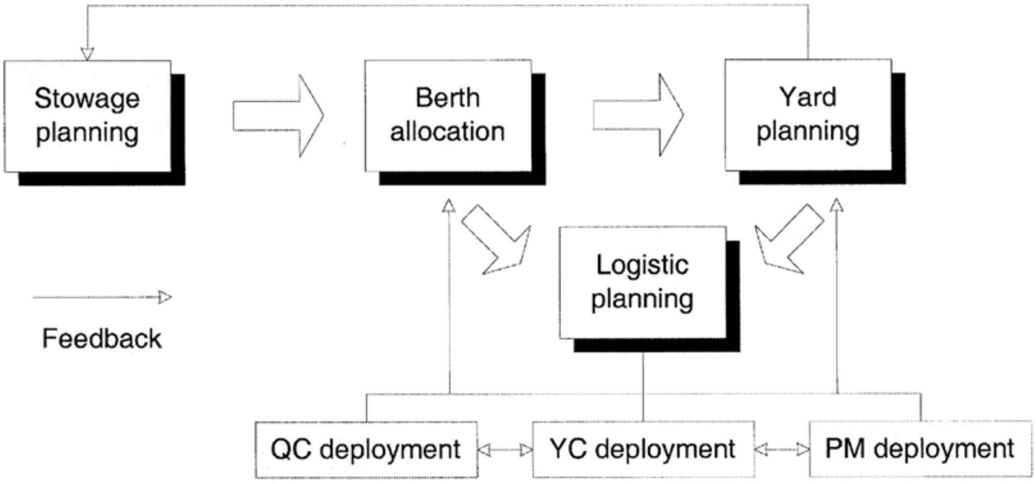

Figure 2.2: Hierarchical structure of decision-makings in container terminals

proposed in the literature to tackle the problems arising in yard operations. On this basis, we attempt to find out the research directions towards the application of simulation and optimization tools and techniques that can be of guidance for the current study.

2.2 Yard planning and crane deployment

Operation on the storage yard, i.e., storage and retrieval of containers, is the most complicated part at a terminal because both inbound and outbound container flows are handled in this area simultaneously. Yard planning hence determines the port efficiency to a great extent. It involves assigning storage locations to inbound containers (i.e., *storage allocation*), sequencing outbound containers, and the deployment of yard cranes.

2.2.1 Stack configuration

In conventional storage yard, containers are stacked by yard cranes side by side and one on top of another to form rectangularly shaped heaps called *blocks*, each of which consists of a number of *rows* in width, a number of *bays* in length and a number of *tiers* in height. The size of each block varies in different terminals, typically with 8 rows and 6 tiers in

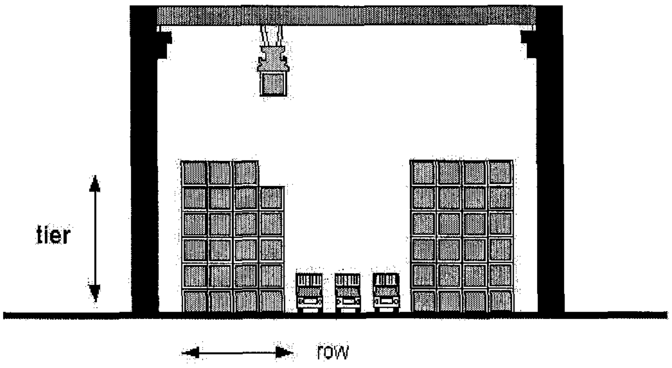

Figure 2.3: A schematic view of an OHBC yard block

the Port of Singapore using OHBC (overhead bridge crane) as shown in Figure 2.3. Obviously one of the problems at the strategic level is to determine a good stack configuration, specifically the height of stacking. It's logical to store containers in higher stacking to save ground space, however the more tires the containers are stacked in, potentially the more reshuffles/rehandles will be required to retrieve a required container.

In [Che99] it was concluded that the higher stacking requires the improvement on all other relevant conditions at the same time, otherwise it would possibly results in a negative impact of causing large numbers of unproductive container movements. Various storage strategies were then described and tested to determine the tradeoff between extra handling efforts for higher stacking and space requirement. [CD93] also studied the stack configuration problem, in which the number of moves to retrieve a container was formulated as a function of stack height and operation strategy, and the best operating strategy could be selected for a chosen configuration. [CRM88] proposed an idea of using a buffer area to store export containers so that the nonproductive movements of yard cranes can be reduced during loading process. A simulation model was built to investigate the effects and 4% reduction in the total loading time was achieved. [Kim97] established a methodology to estimate the expected number of rehandles to pick up an arbitrary container and the total number of rehandles to pick up all containers in a bay for a given initial stacking configuration. Regression equations and an approximation formula were given for evaluation of the number of rehandlings to aid decision making on stacking configuration.

The problem of stacking configuration in an AS/RS yard is converted into the optimal design of storage rack layout, which involves the evaluation of the expected cycle time under different circumstances. Therefore, the existing studies on stacking configuration are not applicable to our case and a brand new scheme is required (see Chapter 4 for details).

2.2.2 Storage allocation

Storage allocation determines the exact locations in the yard for containers, which is a very critical issue ultimately determining the amount and distribution of workload in individual blocks. The allocation usually aims at minimizing workload for loading and unloading processes and is tailored for different types of containers, i.e. *export, import,* and *transshipment*, which are identified with respect to their origins and destinations as follows:

- *Local import* containers: inbound containers from vessels at predictable times, to be removed by external trucks at unpredictable times.
- *Local export* containers: inbound containers from external trucks at unpredictable times, to be loaded to vessels at predictable times.
- *Transshipment* containers: discharged from mother vessels and to be loaded to second carriers, both at predictable times.

Taking into account the predictability/unpredictability of container arrival and departure times, in practice various storage strategies are adopted to disperse containers in different terminals according to the proportions of different types of containers to be handled. For example, due to the high percentage of *transshipment* containers and the limited storage space, operators in Singapore terminals mix *local export* and *transshipment* containers at the block level by grouping them with certain considerations, e.g., their second carriers, and place containers directly into yard blocks that are near the quayside upon their arrivals. While *local import* containers are segregated according to their importers and stacked in an area close to the gate. Accordingly, existing studies on storage allocation also fall into similar categories that serve different types of containers. However, the *transshipment* containers are regarded as *import* containers during their unloading phases, and as *export* containers during loading phases.

Export containers

[TICD93] gave a description of handling efforts under different storage allocations for *export* containers and quantified their performance according to the amount of space and the number of moves required. The strategy that puts containers in permanent positions upon their arrivals was found to offer light workload however suffer almost 50% waste of space. [KB98] discussed how to re-marshal *export* containers in the storage yard to create empty space, which was decomposed into three sub-problems: the bay matching problem matching a specific current bay with a bay configuration in the target layout; the move planning problem determining the number of containers to be moved from one bay to another; the task sequencing problem settling the sequence of container moves. The first two problems were solved by dynamic programming techniques while the third one was formulated as a transportation problem. In [KPR00] the weights of *export* containers were taken as an extra consideration when doing storage allocation. A dynamic programming model was formulated to minimize the number of relocation movements expected for the loading operation and a decision tree was given to support real time decisions. The percentage of wrong decisions, which is a measure of the quality of the decision tree, was found to be less than 5.5%. A generalized approach for dynamic allocation of *export* containers was proposed in [KP02], in which the problem was formulated as a mixed integer linear programming model to maximize space utilization and to minimize loading time. Two heuristic algorithms were suggested to find the solution based on the least duration of stay (DOS) rule and the subgradient optimization technique, respectively. The DOS-based decision rule was found to be attractive as it delivers same quality of results as the subgradient approach while consuming much less computational time.

Import containers

[KK96] investigated the storage allocation problem for *import* containers by segregating them according to their arrival times, with stack height and allocation space as decision variables. The authors extended their findings to cases where the arrival rate of import containers is either constant, cyclic, or dynamic in [KK99a]. It was derived that in the case of constant arrival rate the optimal height of stacks is the total number of containers within the planning horizon divided by the total number of ground slots in the stack. No similar formula was found for the other two cases, but the problem of space allocation was found to be solvable using subgradient optimization techniques. [Rou96] developed simple analytical expressions to estimate the minimum storage capacity needed to ensure infrequent episodes of storage congestion, which also served to identify the effects of changing throughput characteristics on the storage space. [Wat91] provided a method to estimate the number of rehandlings in an import container yard, which leads to an optimal loading/unloading approach.

There are several studies that consider different types of containers as a whole. [CH99, CCH00] developed a time-space network to assist in assigning containers to storage locations in advance with an objective of minimizing total costs of operation. The problem was formulated as a two-dimensional packing problem in the sense that a container at the yard can be represented by a time-space rectangle whose height and width are associated with the stay duration and the space occupation. And a *branch and beam* algorithm was proposed to solve it. [IMP$^+$] applied *eco-problem solving* and object oriented paradigms to build a multi-agent model that facilitates the evaluation of allocation policies. [HVJD99] took into account the *intrinsic* and *logistic* values of containers and divided them into different priority classes, for each of which the optimal amount of space and price were determined under welfare and profit maximizing rules. The problem was more or less formulated as a demand-supply balancing problem, where demand was introduced by arrival rates, price elasticities and logistic opportunity costs, while supply was brought in through marginal operating costs and land requirements. With a rolling horizon approach, [CLW$^+$03] decomposed the allocation problem into two levels and each level was solved by a linear integer programming model. At the first level the total number of containers in each block during the planning horizon was determined to balance the workloads among blocks, while the second level distributed containers to individual blocks for each vessel such that the total travel distance of containers was minimized.

It is found that existing research tends to consider the *transshipment* containers as *import* containers during their unloading phases, and as *export* containers during loading phases. The advantage of predictable arrival and departure of *transshipment* containers is not studied and utilized. Due to the high percentage of *transshipment* containers in the Port of Singapore that we are studying for, a new storage allocation policy should be specially designed for them. However, the objectives of balancing workloads and minimizing travel distances are still applicable and desirable.

2.2.3 Retrieval sequencing

Containers in some hub terminals are normally grouped into several categories by their connecting vessels, destination ports, sizes, contents, and weight classes (in this order, named as vvPSCW grouping in the Port of Singapore). In this case, when a QC asks for a container from a certain category instead of a specific container, a choice can be made among the containers in different blocks taking care of the planned workload of the yard cranes to improve port performance. This is the sequencing problem during loading process that, in its entirety, actually considered crane allocation and routing problems, which is also referred to as the dynamic crane deployment. [ZWLL02] managed to minimize the total delayed workload in the yard by optimally deciding the times and routes of crane movements among blocks. The problem was formulated as a mixed integer programming model and solved by Lagrangian relaxation technique. A modified Lagrangian relaxation was also proposed to reduce computational time that leads to a near-optimal solution. [KP99] used genetic algorithms to schedule the retrieval of containers so that the sum of setup times and travel times, and consequently the ship turnaround time, was minimized. Neural networks, tabu search, or other heuristics were also suggested in [KP99] as possible competing techniques. There are also some studies that covered other aspects with impacts on the loading process such as the packing of containers, see for example in [CLS95, DB99, Sch99]. To the best of our knowledge, the number of papers that focus on the full scope of container loading/unloading process decisions at operational level is very limited. It would be the most useful to look into this area so as to fully understand the impacts of interactions among the decisions. However, it appears that the problem would be too complicated to be solved by analytical methods and the solution heavily depends on the experience of operators.

2.2.4 Crane deployment

There are several kinds of yard cranes used in container terminals to store and retrieve containers in and from the stack, among which rubber tyred gantry cranes (RTG) and rail-mounted gantry cranes (RMG) are most commonly chosen. An RTG moves on rubber tyres and is able to move among blocks. An RMG runs on rails normally serving a single storage block between the rails. Both RTG and RMG provide higher density storage and shorter cycle time than other cranes such as straddle carriers that carry containers between their legs, mobile harbor cranes that are inherited from pre-containerized era, and heavy-duty forklift that are more used for empty containers. Another kind of yard crane, OHBC (overhead bridge crane) with even higher density storage and shorter cycle time is used in the Port of Singapore.

One of the decisions to be made at the tactical level is to determine the number of cranes necessary to ensure an efficient storage and retrieval process. [KK98] discussed this problem with regard to space requirement for import containers. For a given amount of space, more yard cranes result in shorter response time for a retrieval request but higher facility investment. In other words, there exists an economic tradeoff between the storage density, the accessibility, the investment, and the level of service. A analytical model was developed to resolve the tradeoff by minimizing the sum of relevant cost components associated with the number of cranes and the amount of space. The model was extended in [KK02] to take care of the operating cost of the cranes and serve the interests of both terminal operators and customers. The minimization of the operation cost was solved with a deterministic model, and a stochastic model was developed to minimize the waiting cost of external trucks and the operating cost of the terminals simultaneously.

At the operational level, the deployment of yard cranes is crucial to the productivity of a storage yard. As a pioneering research on this problem, [LL94] used simulation to investigate several yard crane allocation policies with throughput, utilization, and waiting times as performance measures. On routing of yard cranes, several studies have been conducted in [KK97, KK99c, KK99b], which considered the single-crane scenario during loading operation of export containers. The container handling time, including the crane setup time and the travel time, was treated as optimization objective and minimized by optimally determining its visiting sequence to the yard bays and the number of containers to be picked up at each yard bay. The loading sequence was constrained to satisfy the work schedule of the corresponding QC that was assumed as an input. The tour of a crane was expressed as a route on a network connected by a series of 'sub-tours', each of which was defined as a sequence of yard bays visited by the crane to pick up all containers that will be loaded together as a cluster onto a ship. In the constructed network, the original routing problem was reduced to finding a path from the source to the sink and to determine the number of containers to be picked up at each node during the tour. It was formulated as a mixed integer program, and was solved by a suggested algorithm that determines the number of containers to be picked up at each bay in the first stage and the route of the crane in the second. It should be noted that the deployment of multiple cranes is hardly addressed in the literature, although it is a much more practical issue. A discussion is given in Chapter 5 where the crane deployment algorithms are proposed for the AS/RS yard.

2.3 Summary

In this chapter, we have presented a survey of the operations at the storage yard of a container terminal and the simulation and optimization issues from a hierarchical viewpoint.

Most of the related studies are found to aim at a particular sub-problem and solve it with analytical modelling approaches. It is evident that in order to simplify the analytical models, necessary assumptions are inevitable. Among the various analytical approaches, mathematical programming, queueing theory, and network-based method appear to be the most popular techniques.

Despite the numerous research done in this field, there are, however, a few outstanding problems that require further study. For example, predictability of arrivals and departures of transshipment containers may allow better storage allocation scheme for the terminals that mainly deal with this kind of containers. Furthermore, some of the assumptions made in some models could be relaxed to be more realistic, whereas some models themselves could be extended to become applicable to a generalized case.

Another finding that is also related to the proposed AS/RS yard is that the deployment of multiple cranes is hardly investigated, which becomes one of the inspirations that drive the current research to get involved in the topic.

Chapter 3

Design and Control of Automated Storage/Retrieval Systems

In this chapter we present a comprehensive survey of the design and control issues in automated storage and retrieval systems (AS/RS). The AS/RS design aspects include the configuration of the storage racks and the optimal number of storage/retrieval (S/R) machines and their speed profiles. The AS/RS control policies cover three elements of scheduling [HSG76]: the assignment of multiple items on the same pallet (*Pallet assignment*), the assignment of pallets to storage locations (*Storage assignment*), and the rules for sequencing storage and retrieve requests (*Interleaving*). Various studies, aiming at improving AS/RS performance, are discussed and compared in different categories, based on which our interesting application of AS/RS in container terminal is motivated and established.

3.1 Introduction

Automated storage and retrieval systems (AS/RSs) are computer-directed storage and transporting facilities for large capacity, high volume material handling. An AS/RS (Fig. 3.1) consists of storage racks with unique or un-unique cells, input/output (I/O) stations for receiving and sending items, and storage/retrieval (S/R) machines with pallets for providing transport between I/O stations and storage cells. Briefly, AS/RSs work as follows: incoming items from the I/O stations are assigned to pallets, sometimes two or more items to one pallet, then the pallet is assigned with a location and stored using the S/R machines;

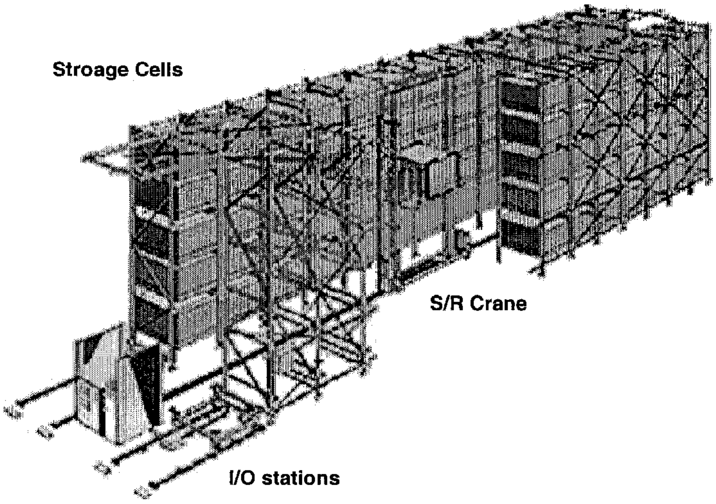

Figure 3.1: A schematic view of AS/RS

upon receiving a request for an item, the S/R machine is directed to retrieve the pallet to the I/O station for the item to be picked up, after which the pallet is returned to its original location for storage, reconsolidation, or re-use.

The effective use of AS/RSs can lead to a number of substantial benefits such as high space utilization, improved material flow, and labor costs. However, the benefits achievable are dependent upon the appropriate *design* and *scheduling* of the system, which can be identified as follows:

(a) *Design aspects*:

  (i) the size of the storage cells, normally decided by the types of items to be handled.

  (ii) the building dimensions and the number of aisles to meet the capacity requirements.

  (iii) the configuration of the storage racks, including the rack size, the positions of I/O stations, and the shape factor that deals with the ratio of the length and height of the racks in terms of S/R machine travelling time.

   (iv) the number of S/R machines and their capacity. An S/R machine that can carry more than one item is able to perform multiple command operation cycle (i.e., storages and retrievals are performed alternatively in a cycle).

(b) *Scheduling aspects*:

   (i) pallet assignment, deciding how many and which items to be stored on individual pallets. Generally in practice, one single item is assigned to each pallet, in which case pallet assignment is trivial.

   (ii) storage assignment, allocating storage locations for incoming items. Three policies are mostly used in practice and research: *randomized* storage, *class-based* storage, and *dedicated* storage.

   (iii) interleaving, the rules for sequencing storage and retrieval requests.

   (iv) dwell point policies, deciding the positions for the S/R machine to stay when idle. Normally such positions are chosen to favor the upcoming requests.

There are numerous measurement criteria for evaluation of AS/RS performance with different focuses that reflect the interests of system operators. To list for instances, some of them are: travel time per storage/retrieval, the number of storgae/retrieval operations performed per unit time, or maximum number of requests waiting in the sequence. However, as a fundamental step in AS/RS design and evaluation, *travel time*, i.e., the expected amount of time for the S/R machine to perform a storage or retrieval operation, is most commonly used in the literature to quantify the improvement achieved by applying their proposed approaches or policies. The throughput of an AS/RS is normally the inverse of the *travel time*, which can be reduced by decreasing the dimensions of the storage racks. However, such decrease will result in the increase in the number of AS/RS aisles to maintain the storage capacity and subsequently the increase in investment. Therefore, it is crucial to estimate the *travel time* that an AS/RS aisle can achieve to satisfy the throughput requirements, while keeping the costs down.

The throughput depends on the characteristics of the demand, e.g., the arrival pattern of requests, the weights and sizes of the items and their dwell durations, and the number of items to be operated on each request. These demand requirements have certain implications on the design of an AS/RS as well as its scheduling. Along with the design and scheduling parameters, they determine the performance of an AS/RS. Due to the number of influencing factors and their interrelation, it is, unfortunately, not possible to incorporate all of them in a single model. As a result, most of the existing studies have been conducted based on various combinations of these parameters for achieving efficient AS/RSs, using various simulation and analytical techniques.

Dated back in early 1960s, the concept of automated warehouse and AS/RS has been proposed and investigated using analytical models with an emphasis on the physical design i.e., configuration or layout of the system. These analytical models make use of assumptions that help in removing or fixing other parameters to focus on some particular aspects most popularly on *travel time*, and produce various results for AS/RS control. On the other hand, simulation has also been another effective tool to explore the suitable combinations of operating policies for achieving high throughput. The simulation models relax the assumptions and therefore are capable of examining more parameters simultaneously The results provided are more practical, though the development of these models is rather time-consuming.

With an attempt of grouping and evaluating all these models on a common scale, the rest of this chapter is organized as follows. Section 2 discusses the studies for the scheduling issues, e.g., storage assignment, interleaving, and dwell point policy. The design aspects and the performance evaluation of AS/RSs are highlighted in Section 3 and Section 4 respectively. Section 5 presents some concluding remarks and points out the promising directions for future study.

3.2 Scheduling of AS/RS

A good scheduling of AS/RS requires a suitable combination of policies for *pallet assignment, storage assignment, interleaving,* and *dwell point selection.* The expected *travel time* of S/R machine in an AS/RS is normally considered as a measure of the AS/RS performance. The improvement of any proposed approaches is also normally quantified by the comparison on *travel time.* Therefore, the estimation of *travel time* for an AS/RS has become a fundamental step towards the optimal design or control in most analytical model-based studies. In most studies, the storage rack is assumed to be a continuous pick face, i.e., each storage location on the rack is considered as a point in an area.

3.2.1 Studies on storage assignment

Incoming items can be stored into the storage rack according to their turnover rate (i.e., the number of times a given item is stored or retrieved per unit time) or to any arbitrary open locations. The turnover-based storage divides the incoming items into a certain number of groups according to their turnover rate, and the rack into the same number of classes, each of which is dedicated for one turnover rate group. The main issue to be resolved is thereafter the optimal division of classes that leads to the minimum expected *travel time.* On the other hand, the randomized storage randomly selects an open location in the rack for each incoming item. It is used to approximate the 'closest-open-location' rule popularly adopted in industries, and serves as a benchmark to other storage policies.

Turnover-based storage assignment

As one of the earliest research, [HSG76] revealed the substance in AS/RS control by outlining the design and operational issues and the interrelationship. It was suggested storage policies to be based on the level of turnover frequency of each item. For a single-command square-in-time AS/RS (i.e., the time to reach its farthest bay is equal to the time to reach the highest tier), two such policies were proposed: full turnover-based assignment and class-based turnover assignment. Under full turnover-based assignment, an item with a higher turnover frequency is stored closer to the I/O station. The class-based turnover assignment divides the storage rack into several classes and the items into the same number of groups, and each storage class is associated with one group according to their demand. A storage allocation policy is a mapping between an item in a group and a storage cell in the associated storage class. Essentially, the full turnover-based assignment is a class-based policy with one item per class. It was found that class-based turnover assignment is easier to implement compared with the full turnover-based assignment, but offers almost the same level of significant potential reductions in *travel time*, which can be expressed as the function of the boundary R in a two-class system [HSG76]:

$$T_2'(R) = \frac{2}{3}\left[R^{\frac{5s+1}{s+1}} + \frac{(1-R^3)\left(1-R^{\frac{4s}{s+1}}\right)}{1-R^2}\right] \qquad \text{(Eq. 3.1)}$$

where s is the cumulative % demand versus % of inventoried items. The derivative of the above equation with respect to R leads to an equation from which the optimal value of R can be derived. The result for three-class system was also presented in a similar way. However, for four and more classes, the procedure for obtaining optimal R was not given in [HSG76].

The generalized solution procedure for finding the optimal boundaries of each rack class was proposed by [RE89], where similar assumptions were made to remove the *pallet assignment* problem and to set S/R machine dwell at I/O station when idle. For an n classes normalized storage rack (i.e., the value of each boundary is divided by that of the last boundary), let R_i be the boundary between the class i and class $i+1$, then the one-way *travel time* T_n of class n was given as a function of R_{n-1} as follows [RE89]:

$$T_n\left(R_{n-1},1\right) = \frac{2}{3}\left[1-R_{n-1}^{\frac{4s}{s+1}}\right]\left[\frac{1-R_{n-1}^3}{1-R_{n-1}^2}\right] + R_{n-1}^{\frac{5s+1}{s+1}} \cdot T_{n-1}(1) \qquad \text{(Eq. 3.2)}$$

In order to find the optimal class boundaries for the n-class system, the $n-1$ class system need to be solved first. Recursively, the solution procedure is a dynamic program with $n-1$ stages, at the kth stage of which $T_k(1)$ is obtained by minimizing the above equation over $0 < R_{k-1} < 1$. The searching procedure is a one-dimensional search and therefore efficient.

The results were further extended for rectangular-in-time AS/RS in [ER94], where shape factor b was introduced and defined as:

$$b = min\left(\frac{t_v}{t_h}, \frac{t_h}{t_v}\right) \qquad \text{(Eq. 3.3)}$$

In this definition, t_v is the one way travel time from the I/O station to the highest tier, and t_h is the one way travel time from the I/O station to the farthest bay. For a normalized n class rack with width of $\frac{1}{\sqrt{b}}$ and height of $\sqrt{b}$, there are three different types of classes that may exist [ER94]:

(a) square classes: "L" shaped classes commonly found in the square-in-time AS/RS, whose boundaries $R_i \leq \sqrt{b}$ form (together with previous classes) the square-in-time area within the rack with dimensions $R_i \times R_i$.

(b) rectangular classes: their boundaries are $R_{i-1} \geq \sqrt{b}$, along with the previous classes forming a rectangular area with dimension $R_i \times \sqrt{b}$.

(c) transient classes: between the square classes and the rectangular classes, whose boundaries are $R_{i-1} < \sqrt{b}$ and $R_i > \sqrt{b}$.

Also in [ER94], an algorithm involving one dimensional search was proposed to find the boundaries for any desired number of classes that consist of only the square classes and a transient class. In addition, a two-dimensional search procedure was given for the case that rectangular classes exist. The *travel time* T_n was found to decrease with the increase in n and b, and the decrease in s.

On the basis on these results, [KP95b] presented explicit formulas for the optimal boundary and the expected travel time for a two-class-based single command AS/RS rack. [PW99] proposed a framework for the dual command cycle travel time model under an n-class-based storage policy. [AHBW01] determined the expected travel time for an class-based AS/RS with two I/O stations using extended geometrical-based algorithm. This is a practica problem-oriented extension of the analytical models. [WCC01] considered the speed profile of S/R machine, i.e., the acceleration and deceleration, and developed analytical expressions for the expected *travel time* of a rectangular-in-time single command AS/RS. An exponential model and an adjusted exponential model were found to satisfactorily approximate the *travel time*. [MK01] investigated the effects of relocation to class-based storage with production quantity variation and suggested that relocation be done when the variation up to certain level. Another interesting study done by [Tho98] applied the turnover-based assignment to the *item* level (rather than *item group* level in a stochastic environment and found that the class-based assignment is practical and efficient when applied to stochastic demand of items. To the best of our knowledge, [Tho98] is the only research related to *pallet assignment* found in the literature.

As the special case of class-based storage assignment, dedicated storage policy essentially store the item with highest demand at the location closest to I/O station, i.e., the distance (in time) of an item from I/O is inversely proportional to its turnover frequency Studies of dedicated storage assignment are commonly found with those of class-based assignment as a comparison, e.g., how much of benefits of it can be gained by a class-based policy [HSG76, RE89, Tho98, WCC01]. However, some research did focus on dedicated storage itself. [Man97] presented a computerized algorithm for investigating dedicated allocation alternatives and the combined and individual impact of space, speed, and storage allocation on the system performance. [MLR98] dealt with the problem of minimizing, instead of the average load normally concerned, the peak load in a single command AS/RS with dedicated storage policies over a fixed horizon. Mixed integer programming model were proposed and their solutions by branch and bound technique were also given. Early studies addressed dedicated storage with an objective function including both average load and inventory costs [HL82]. It was developed to a so-called duration of stay (DOS)-based storage assignment in [GR69] based on industry practice, in which the unit loads having the shortest expected DOS value were stored in the closest cells. It operates based on individual unit load characteristic rather than the average product turnover rates. A comparison of performance of these two policies was done in [KOSB99] that concluded that the turnover-based assignment generally outperforms the DOS-based policy.

Randomized Storage

The closest-open-location rule was often used in practice before turnover-based assignment was proposed, which always assigned the item to the closest open location. For analysis purpose, it is normally approximated by randomized storage assignment proposed by [HSG76], in which any item is equally likely to be stored in any of the open locations within the rack. The randomized storage is popularly used as a benchmark to other storage policies, and the expected one way travel time under this policy was found to be 2/3 time units for a square-in-time rack [HSG76]. To extend the continuous approximation of *travel time* for a rectangular-in-time rack, [Boz84] developed analytical models for both single command and dual command conditions using a statistical approach, which is discussed in details below:

The accurate value of the expected *travel time* (in a discrete rack as the real case) can be computed as follows:

(a) expected single command *round trip* travel time [Boz84]:

$$E\left(\overline{SC}\right) = \frac{1}{N}\sum_{i=1}^{N} 2t_{0i} \qquad \text{(Eq. 3.4)}$$

(b) expected dual command travel time [Boz84]:

$$E\left(\overline{DC}\right) = \frac{2}{N(N-1)}\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\left[t_{0i} + t_{ij} + t_{j0}\right] \qquad \text{(Eq. 3.5)}$$

where N is the total number of openings in the rack, t_{0i} is the one way travel time between the I/O station and the ith opening ($t_{0i} = t_{i0}$), t_{ij} is the one way travel time between the ith opening and the jth opening, and t_{j0} is the one way travel time from the jth opening back to the I/O station.

Assume that the coordinate locations are uniformly distributed, the expected travel time of the normalized continuous rack under single command $E(SC)$ is obtained as follows [Boz84]:

$$E(SC) = \frac{b^2}{3} + 1 \qquad \text{(Eq. 3.6)}$$

where b is the shape factor (definition given in previous paragraphs). And the expected travel time for a complete dual command cycle $E(DC)$ is derived as [Boz84]:

$$E(DC) = \frac{4}{3} + \frac{b^2}{2} - \frac{b^3}{30} \qquad \text{(Eq. 3.7)}$$

These continuous models were proved to perform in a satisfactory manner with the largest percentage deviation being 0.2069%. It was also observed that a square-in-time rack minimizes *travel time*.

To relax the assumption of constant S/R machine velocity, [HL90] took into account the operating characteristics of the S/R machine, including the acceleration and deceleration rate and the maximum velocity restriction, and modified the travel time models from practical perspectives. [CWL95] proposed additional amendment for the travel time models in randomized storage by considering the speed profiles of the S/R machine that exist in the real world application.

Another relaxation is AS/RS with un-unique cell, which is necessary to meet the requirements from the changing business environment. [LTC99] developed two *travel time* models for racks with un-unique cells under randomized storage assignment.

3.2.2 Studies on interleaving rules

Besides performing a single storage or retrieval operation between the I/O station and a storage location, the conventional S/R machine with one shuttle to transport loads is capable of visiting up to two rack locations between successive returns to the I/O station such that upon completing a storage request, it moves directly to the rack location for the next retrieval without returning to the I/O station. Obviously, such operation cycle implies potential higher throughput due to the shorter travel distance, if the storage and retrieval requests are properly paired. Therefore, the interleaving rules, i.e., the sequencing of storage and retrieval requests, play an important role in improving AS/RS performance by working closely with the storage assignment policy. It is henceforth not surprising that the searching for an optimal combination of storage assignment and interleaving policy has been the subject of numerous studies.

The most common way of interleaving incoming requests is to pick individual retrieval requests to pair with the storage requests so as to achieve performance improvement (i.e., *alternating retrieval and storage requests*). Such technique could be further improved by using multi-shuttle stacker cranes and optimally scheduling crane tasks (i.e., *multi-shuttle technology*). Another way of interleaving is to batch incoming requests such that the travel distance of the S/R machine to finish the desired operations is minimized (i.e., *batching retrieval and storage requests*). These techniques are discussed in the following.

Statically alternating retrieval and storage requests

[GHS77] examined several alternative combinations of storage assignment and interleaving policies based on the analytical travel time models, whose adequacies are tested by discrete evaluation. The performances of a system allowing interleaving and a system disallowing interleaving were investigated under four types of storage assignment (randomized storage, two-class-based, three-class-based storage, and dedicated), and two types of interleaving rules (first-come-first-served, referred to as FCFS, and selection queue of K retrievals). The results indicated that the worst scheduling policy is RAN/NIL/FCFS, i.e., randomized storage assignment, no interleaving, FCFS retrieval queue. However, this policy could be improved by adopting class-based storage, by adopting interleaving, or by both towards the best performance obtained by FULL/MIL/$K = 5$ (dedicated storage, mandatory interleaving, selecting retrieval from a queue of 5). The rule for the selection of retrieval from among the queue was: after completing a storage, the first K items in the retrieval queue are sequentially examined for a retrieval of the same class; if none of such item is found then the first retrieval in the queue is performed. This rule, however, is applicable to class-based storage only.

Dynamically alternating retrieval and storage requests

Taking into account the dynamic nature of the changes of retrieval requests, [HMSW87] proposed a nearest-neighbor sequencing rule to select retrieval from the queueing list, and developed an analytical model for its expected throughput performance, whose solution was found through Monte Carlo simulation. The results showed that the proposed nearest-neighbor sequencing is easy to implement and able to reduce travel time between two consecutive operations by 60% for the list containing 15-20 retrievals, compared to FCFS rule. And for a typical dual command AS/RS, this 60% reduction in travel between times could yield a 12% increase in system throughput. The model is applicable to randomized storage only.

Instead of dealing with large scale AS/RS, [KM88] proposed a similar policy called closest to next location storage for mini-load AS/RS, under which the same type of items are stored at the empty locations which are closest in time to their next (or first) workstation, and the retrieval is selected from the location that is nearest in time to the S/R machine. It was found from the analysis that this policy outperforms class-based storage when the inventory level is low. For the systems with inconsistent and/or high inventory level, the number of eligible locations is small and therefore it may simply work as a class-based storage without sequencing rules.

Similarly, [MRP98] formulated an easily implementable nearest-neighbor strategy for mini load AS/RS that takes a batch of retrieval requests and sequences it appropriately. An analytical model was developed to predict its performance, which was validated using simulation. Two steps were involved in the proposed heuristic, the first step rearranged the retrieval requests in close proximity to each other as measured by the Tchebyshev distance metric, whereas the second step interleaved the output from the first step to combine items into a dual command cycle. The results showed 5-15% improvement over FCFS sequencing. It was found that a tradeoff exists between the level of reduction in travel time and the required sorting time.

Targeting at serving both unit-load and mini-load systems under dedicated storage assignment, [LS96] presented static and dynamic approaches to sequence the retrieval requests, the former one performing interleaving in blocks, whereas the latter one reflecting the arrivals of new requests. Heuristic methods were proposed to obtain near-optimal solutions to the approaches, and it's found that the static sequencing increased the throughput by up to 9%. The dynamic heuristic was simpler and faster in terms of computational time, and offered even more increase in throughput of at least 14%. The approaches were found to be applicable to randomized storage as well, but long delays may occur for certain retrieval requests, in order words, the tradeoff between efficiency and urgency was ignored. Furthermore, the static heuristic proposed may lead to an infeasible sequence since a location may be chosen for storage even when the item had not been retrieved yet.

Being aware of this, [vdBG99] identified the empty travels that may occur in an optimal sequence and incorporated them in a transportation problem for the dedicated storage system, which makes it possible to find an optimal solution in polynomial time in static sequencing. Two heuristics were designed that lead to near-optimal results on average. A single command area was also identified within the rack, where a dual command cycle is never shorter than two single command cycles. The decrease of such area was observed to improve the performance of the proposed algorithm.

Multi-shuttle technology

The throughput of an AS/RS could be further improved by multi-shuttle technology, which, e.g., a twin-shuttle system, increases the carrying capacity of the S/R machine to two unit loads and hence enables four-command cycles. In a typical four-command cycle with randomized storage, the S/R machine carries two loads from the I/O to the first storage location, places a load into it, and then moves to the first retrieval location, retrieves the load from it, and places the second load into the vacated rack position, following that it travels to the second retrieval location, retrieves the load, and returns to the I/O station with two retrieved load. The travel distance of the S/R machine in a four-command cycle is apparently much shorter than four single command cycles that offers potential throughput advantage. However, the improvement of throughput is again dependent on the storage policy. Several studies are found that deal with the interleaving of dual shuttle AS/RS as follows.

[SSLH91] presented a retrieval sequencing heuristic along with an approximation for the expected command cycle time to compare the throughput of a dual-shuttle four-command system with that of a single-shuttle dual-command system under nearest-neighbor approach. The results indicated that the improvement varied from 53% to 82% depending on the number of open locations and the number of retrieval locations for a given number of open locations. It was also found that simply applying a class-based heuristic to a dual shuttle system will not guarantee optimal results because the more possible routings associated with a four-command cycle in randomized storage may offset the advantages of the dedicated storage policies. The related dispatching and performance issues were investigated in [KP94] by simulation models.

Analytical models to estimate the interleaving rules to improve the throughput of multi-shuttle (more than two) AS/RS were developed in [MM97], which proposed nearest neighbor rule to achieve 20% throughput increase for a triple-shuttle AS/RS.

The dynamic behavior of the queue of storage and retrieval requests was addressed and described by an state equation-based analytical model in [Mal00]. Using the estimations of travel times for various cycles associated with the demand levels of the requests, the state equations were applied to estimate the probability distribution of the queue states, which was used to approximate the system performance. The model could aid designers in eliminating unpromising system configurations prior to undertaking more detailed simulation-based studies. Unfortunately, the model was implemented only for a simple problem, the full implementation and the computational optimization of the model still remain unsolved.

Batching retrieval and storage requests

Other than picking individual retrieval requests to pair with the storage requests, there is another way to achieve optimal interleaving by grouping two or more requests into batches such that the travel time for S/R machine to finish the desired operations is minimized. This is called order batching, on which only a limited amount of research can be found out of the rich literature dealing with AS/RSs. [HL88] formulated the problem as a modified vehicle routing problem using set partitioning concept and presented several heuristic algorithms to find the optimal solution based on cluster analysis. Orders were batched according to the value of the similarity coefficient defined in terms of attribute vectors. The results indicated that order batching is also a NP-hard problem and the proposed hierarchical and sequential clustering algorithms performed well in finding a near-optimal solution. [EU89] provided a number of algorithms to evaluate the travel time of the grouped orders for each batch so that the best batching could be found. The travel time model was extended from [Boz84] by including as the other parameter the number of locations to be visited within one tour. The basic idea of the algorithms was that from among all (or some, which were defined as large orders) of the possible combinations, searched for the group of orders with a maximum saving value in travel time. The due date constraints of items were taken into account in [Els91], where the single command cycle and dual command cycle were both considered when batching the orders. The storage orders were sequenced with maximum longest travel time and earliest due date rules and included in batches with nearest location rule, whereas the retrieval orders were batched by nearest schedule, shortest processing time and most common location rules. Different combinations of the rules were tested and it was found that the nearest schedule for retrieval order batching outperformed others. However, the global optimum was not achieved because only limited rules were considered.

3.2.3 Studies on dwell point strategies

When considering optimizing interleaving of requests, it is assumed that the request list is known beforehand. However, there are such situations that no requests are available, or that the requests arrive in a non-dense frequency. In this case, the S/R machine will travel to its dwell point position, i.e., the location for the S/R machine to reside when idle. An optimal dwell point strategy will also improve the throughput performance of the system by minimizing the service *response time* for incoming requests, and henceforth has attracted the interest from various researchers.

[GHS77] intuitively selected the dwell point at the I/O station. The optimality of this strategy was proven by [Par91] to the case that the probability of the first operation after an idle period being a storage is at least $\frac{1}{2}$. More flexibly, [Boz84] considered the following four fixed dwell points for different AS/RS configurations for the I/O stations, under randomized storage assignment:

(a) return to the input station after a storage job; remain at the output station after a retrieval job.

(b) remain at the storage location after a storage job and at output station after a retrieval job.

(c) travel to the midpoint location in the rack after a job.

(d) travel to input station after a job.

It was found that strategy (b) performs best in a square-in-time continuous rack in most of the cases. It is, however, not necessarily the optimal dwell point because the expected travel time is not minimized. To find the optimal dwell point from the continuous rack, [PSH96] used a decomposition approach to obtain the expected *response time* and developed an algorithm to derive the optimal solution corresponding to the optimal dwell point. [Par99] developed an optimal dwell point policy for dedicated storage taking as parameters the skewness factor of the pallet turnover distribution and the probability of the incoming request type. The Input station was found to be a good alternative dwell point for dedicated storage. For rectangular-in-time rack, [Par01a] presented closed form solution in terms of the probability of the next operation request type and introduced various effective return paths to the dwell point for the case that the S/R machine's move towards the dwell point could be interrupted by an incoming request. Covering both randomized and class-based storage assignments, [vdB02] modelled the dwell point problem as a facility location problem with rectilinear distance and developed analytical expressions for the optimal dwell point positions. The advantage of this approach is that it can be easily adjusted to accommodate other configurations.

Due to the dynamic nature of the incoming requests, it is apparent that a dynamic, rather than fixed, dwell point should be more suitable for S/R machine control. [Egb91] addressed this issue and developed two LP-based mathematical programming models that respectively minimize the expected and maximum response time of the S/R machine when originating from the dwell point to the demand destination locations. A new dwell point was set whenever it was idle, by simultaneously considering both of the position of the machine on the linear track and that of the shuttle. The dynamic selection and the two-dimensional location view point in effect made the models outperform the above mentioned fixed dwell point rules [EW93]. When the S/R machine is dedicated to serve an arbitrary number of aisles in an AS/RS, the dwell point becomes more critical because it will travel more distance to resume its work from idle status. [CE97] assumed a dynamic order in which the requests arrive over time and determined a dwell point for the S/R machine whenever it becomes idle by a mathematical programming model in which the problem was viewed as a three-dimensional location problem deciding both the grounding position for the machine and the height for the retrieval arm of the machine.

3.3 Towards optimal design of AS/RS

The physical design of AS/RS is a critical issue that on one hand reflects the feedbacks from optimal scheduling and on the other responds to the requirements from the actual application, e.g., storage capacity and cost constraints. An optimal design of AS/RS at the tactical level would help in fixing the the range of such basic features as storage rack height and depth, the number of storage aisles and building dimensions to support decision-makings [Mal01]. As stated in Section 3.2, the optimal design of AS/RS configuration involve a number of aspects, e.g., the height and depth of the rack and their ratio measured in time, the capacity and the number of S/R machines, and the locations of the I/O stations . In practice, some of these factors are addressed as a supplementary result of a study. For example, the optimal shape factor for randomized storage rack was suggested by [Boz84] in building the travel time model. To the best of our knowledge, relatively limited number of studies have been done particularly for the physical design issue, some of which are discussed in this section.

[Zol82, Zol96] introduced a number of variations of heuristic rules of thumb to predict the S/R machine utilization and in effect produced a wide range of operating scenarios and rack configurations for the purpose of preliminary system design. The expected S/R machine utilization were computed based on the machine speeds, the dimensions of the rack, and the demand rate of the incoming requests as follows:

$$U = \frac{\lambda_s + \lambda_r}{60 \cdot a} \cdot t \times 100\% \qquad \text{(Eq. 3.8)}$$

where U is the expected S/R machine utilization, λ_s and λ_r are the total storage and retrieval requests per hour, respectively, a is the number of storage aisles, and t is the expected travel time in minutes. With this formulation, given a target utilization, a cost effective design could be identified through searching over different possible combinations. The model provided intuitiveness, easy application, and in most of the cases, an acceptable accuracy. However, it heavily depends on the conditions that the proportion of single and dual command cycles and the total storage capacity are known beforehand.

In order to relax these assumptions, [Mal01] proposed a modified version of this model in which state equations for the system were given to estimate all feasible flows. Based on these, the state probability for each aisle was derived to estimate the first condition, and a simulation-based sampling procedure was given to estimate the second condition. Efficiency was demonstrated but the computational time could be, however, further improved by other optimization tools such as genetic algorithm, which was considered as a direction of further study.

Also aiming at finding optimal rack configuration, [CW97] presented an analytical model that takes S/R machine speeds as parameters to minimize expected travel time. The results indicated that the optimal rack for the single command cycle is square-in-time whereas that for the dual command would depend on the characteristic parameters. The model was built under randomized storage with FCFS sequencing rule.

3.4 Performance evaluation of the combination of different policies

Performance analysis of an AS/RS is a complex problem due to the fact that the AS/RS performance is sensitive to a number of parameters related to its design specifications and control policies. Among them, the impacts of various control policies could be comfortably evaluated in terms of S/R machine *travel time* by analytical models with certain assumptions as described in the previous section. Some of the design issues, e.g., the *shape factor* of a storage rack, could also be addressed analytically. However, when considering the combined effects of all the interrelated parameters, it becomes evident that simulation is mandatory to adequately model all the features of the system, especially the design factors, e.g., the number of S/R machines.

There are various performance measures for AS/RS adopted in different applications, among which the most commonly found in the literature can be classified as follows:

(a) *Throughput* related, the number of requests processed in a unit time interval, also measured the expected travel time, or mean response time to storage/retrieval requests.

(b) *Resource utilization* related, the proportion of time that the resources, e.g., S/R machine are busy.

(c) *Request delay* related, the mean time that the requests spend in the queue before being processed. It is sometimes measured by the length of queue.

In simulation studies, besides the three types of storage assignment policies, the interleaving rules are organized into rules for selecting requests and rules for selecting open locations for storage requests, which can be elaborated as follows:

(I) rules for selecting requests from the queue [Ber00].

(i) first-come-first-served (FCFS), the requests are executed in their arrival order.

  (ii) longest waiting retrieval (LWR), alternates storage and retrieval requests and serves them according to their arrival times.

  (iii) nearest retrieval (NR), alternates storage and retrieval requests by selecting storage requests according to their arrival times, earliest first, and retrieval requests according to their distances to the S/R machine, shortest first.

  (iv) nearest storage/retrieval (NSR), selects the request whose starting position is closest to the S/R machine, the I/O station being considered as the starting position of a storage.

(II) rules for assigning open locations to storage requests.

  (i) random open location (RN), randomly selects an open location [HSG76].

  (ii) closest open location (CL), selects the open location closest to the I/O station [Ber00].

  (iii) nearest neighbor (NN), selects the open location closest to the following retrieval position [HMSW87].

  (iv) shortest leg (SL), selects an open location such that the travel distance from the I/O station to this location and then to the following retrieval position is minimum [HMSW87].

Based on these policies, a number of studies have been conducted to search for the optimal combination of them with design aspects. [RS95] compared six different layouts of an AS/RS under three different scheduling policies with system throughput as the primary performance measurement, and request waiting time and request reject rate as supplementary criteria. The results showed that the interaction between layouts and scheduling policies was significant at significance level of 0.01 to 0.05, and that for all the scheduling policies considered, the square-in-time rack with class-based assignment performed the best, whereas the combination of FCFS and NR rules produced better throughput for all layouts examined. On storage/retrieval waiting time, the CL/NR combination was found to be the best.

[Ber00] presented a simulation study of an AS/RS with various elements of control policies showing both the isolated and combined effects of these policies, based on which the most suitable policy can be selected for specific cases. When taking into account the space requirement, the class-based storage with NN rule for selecting open locations seemed to be most competitive. For selecting open location, the NN rule was found to perform best; for sequencing the requests, NSR and NR both reduced the expected travel time at the cost of considerable long waiting time for some retrievals, which suggested that if due date performance is the most important requirement, LWR should be applied. The tradeoff between throughput performance and due date performance was consequently addressed, however the optimal combination of policies was not given as a result. The study also proposed a new storage assignment, continuous assignment, to take the advantage of short expected travel time offered by the dedicated storage and that of low space requirement by randomized storage, and proved its efficiency.

Other than the performance evaluation of AS/RS, the modelling technique itself was discussed in some papers that consider AS/RS and its interface with external system as a whole and proposed module-based simulation approach taking AS/RS construction cost into account [Mul89, Tak89, Tak93, Tak94, Tak96, GGB93].

3.5 Summary

In this chapter a review was given to discuss the efforts that have been made to improve AS/RS performance from both design and operation perspectives. It was found from the literature that with certain assumptions, AS/RS behaviors can be comfortably modelled by analytical methods that take care of particular issues arising from various interests of research. The continuous approximation of the discrete storage rack was efficient in estimating the system performance measured by expected S/R machine *travel time* responding to different distributions of request arrival patterns. However, due to the considerable number of influencing factors and their interrelationship, experiment-based study seems to be a better way to look into the combined effect of these factors and to search for the local optimal solution, although numerous studies have been done to optimize individual parameters.

Despite the significant amount of research done to identify and investigate the impact of decision-makings on the performance of an AS/RS, some problems are, however, remaining outstanding or partially unsolved. For examples, the optimal number of S/R machines with respect to productivity and operation costs has not been fully studied, neither the pallet assignment problem. Furthermore, the level of discreteness of the storage rack may affect the accuracy of its continuous representation, which requires further investigation.

On the other hand, the claimed optimizations of AS/RS control are rather sensitive to the demand factors, e.g., the probability distributions of arrivals, and the locations of the requests. Therefore, for any specific AS/RS applications in industry, a detailed corresponding study becomes inevitable and essential to evaluate the applicability of the existing or any newly proposed approaches. And the theoretical results may work better if coupled with field expertise towards the common objective of improving the system performance. This is exactly the reason why we should carry out an independent study when applying AS/RS in container storage with previous results as valuable references.

Chapter 4

AS/RS for Container Storage

In this chapter, we introduce the concept of applying AS/RS in container storage to meet the ever increasing throughput demand on container ports. In particular, we develop the travel time models for the proposed system and confirm its feasibility in terms of throughput performance. Additional constraints arising from the container size and weight on AS/RS design and scheduling are taken into considerations, which in effect leads to a Split-platform AS/RS (SP-AS/RS in short).

4.1 Introduction

Stacking containers side by side and one on top of another suffers a low productivity and ground space utilization that appear to be the bottleneck of current container port operations and become an obstacle to satisfy the ever-increasing throughput demand with limited land availability. A storage structure that offers random access to storage locations would be meaningful to tackle the inherent problems caused by the conventional stacking scheme. This has been the main motivation of the concept of applying AS/RS for storing and handling containers.

Compared with other types of cargos handled by AS/RS in industry, sea containers are specially large in size and heavy in weight, which imposes additional mechanical and safety requirements to their handling systems. Stacker cranes used by conventional AS/RS are not adequate for handling heavy loads at a high turnover rate in container application. Therefore a split-platform scheme is proposed, where transports of the containers within

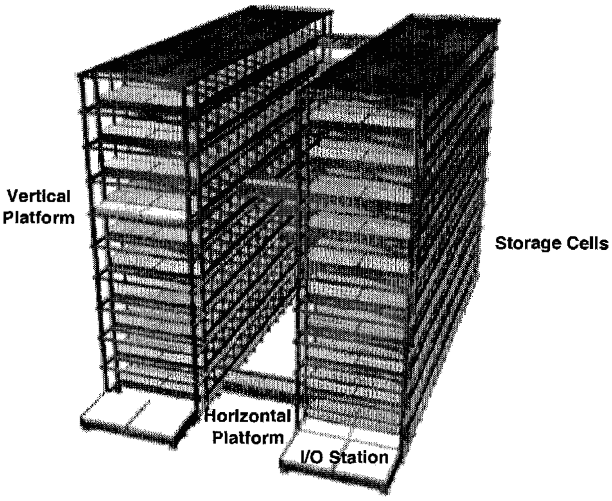

Figure 4.1: Major components of a storage aisle in SP-AS/RS

individual storage aisles are separated into vertical and horizontal movements and handled by different devices, namely the vertical platform (VP) and the horizontal platform (HP), respectively [CHH$^+$03]. Figure 4.1 gives a schematic view of a standard aisle in this proposed AS/RS. The I/O stations are the interface with the external system that carries loads to/from the AS/RS, and the hand-over stations are the locations where a loaded VP delivers the container to an empty HP or vice versa. The VPs transfer loads in between I/O stations and the hand-over stations at any tier of the storage racks, whereas the HPs provide the horizontal connection from the hand-over stations to the individual storage cells. Such a system is capable of concurrent operations, that is, the VPs and the HPs can move independently and in parallel, and henceforth brings the potential benefits of higher handling rate, easier maintenance, and reduced down-time.

The storage operations are performed in the SP-AS/RS as follows, and the retrieval operations are just reversal of the sequence:

(1) The VP moves from its dwell position to the I/O station to pick up the load, then lifts it to the hand-over station on the destination tier; meanwhile, the HP at the corresponding tier moves from its dwell position to the hand-over station.

(2) The load is transferred from the VP to the HP at the hand-over station.

(3) The HP carries the load to the destination storage cell and returns to its dwell position afterwards; at the same time the VP travels to its dwell position if no new jobs arrive.

It is obvious that the proposed system works in a completely different fashion than the conventional AS/RS with combined single S/R mechanism. As such, conceptualizing tools are needed to support the physical design and performance feasibility decisions for the SP-AS/RS. This chapter addresses the main issues in SP-AS/RS design and control, i.e., rack configuration design, storage assignment, and interleaving rules. In particular, it develops travel time models for the system under randomized storage, which take as parameters the basic system attributes similar to those associated with conventional AS/RS. Based on the analysis an optimal layout of the system is proposed, and a dwell point policy is chosen from among several alternatives. In addition, several operation scheduling algorithms are discussed that can significantly improve the throughput performance of SP-AS/RS.

4.2 Discrete travel time models

Travel time in the AS/RS context refers to the expected time required for the platforms to complete an operation in the storage rack. Under randomized storage assignment, the probability of accessing any storage cell is identical. Therefore, the straight-forward way of deriving the expected travel time of an AS/RS is to calculate the travel times to individual storage cells and take the average of the sum.

Consider a single rack with B bays and T tiers to aggregate a capacity of $N = T \times B$. Its I/O station is located at the lower left-hand corner, and the hand-over stations are at the 0^{th} bay of each tier. Let (b, t) denote the cell location at bay b and the tier t. We would calculate the travel time for an access to position (x, y) assuming that the location of the immediately preceding operation is (x', y'). The definitions are illustrated in Figure 4.2.

By the above definitions, we have the following observations:

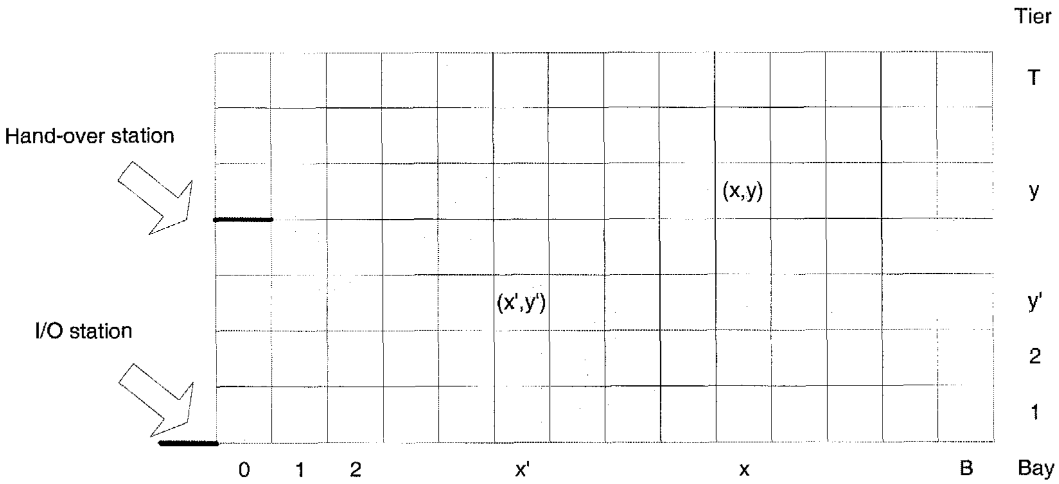

Figure 4.2: Definitions of open locations

Property 4.1 *If* d_h *denotes the dwell position of the corresponding HP, and* d_v *denotes that of the VP, then we have* $d_v = y'$ *and* $d_h = x'$ *following a storage, and* $d_v = 1$, $d_h = 0$ *following a retrieval.*

The main assumptions for the analysis are as follows:

(1) Randomized storage is used, which means that any empty cell with the rack is equally likely to be selected for storage and any occupied cell is equally likely to be selected for retrieval.

(2) The system operates on single command basis for unit loads.

(3) Dwell point policy for platforms is that they will stay where they are at the end of each operation.

(4) Each operation in the infinite sequence of operations is independent of the previous operations, provided that if a storage operation is done on a cell, the next operation to the same cell will be retrieval. The same applies to a retrieval operation.

(5) The VP takes one time unit to travel one tier, and the HP takes one time unit to travel one bay.

(6) Pick-up, deposit, and load transfer times associated with load handling are ignored.

(7) Specifications of the rack and the platforms are known. The platforms travel at constant speeds.

(8) There is no concurrent movement of platforms for different operations.

4.2.1 Storage time model

Consider a storage operation in a balanced system, its cycle time is dependent on the preceding operation only. In an infinite request sequence consisting of both storage and retrieval, any operation except the very first one is preceded by either a storage or a retrieval with equal probability, i.e., 0.5. The storage to cell (x, y) could be decomposed into two phases:

(1) The VP moves from tier d_v to tier 1 and after picking up the load, moves to tier y; meanwhile, the HP at tier y moves from d_h to the handover station at $(0, y)$ for the transfer of the load.

(2) After the transfer, the HP at tier y moves to cell (x, y).

Property 4.2 *For each phase, the completion time is decided by the maximum travel time of VP and HP according to the ability of concurrent movement of platforms for a particular operation.*

Note that the probability of accessing any cell is equally likely, there are two cases to be considered:

(a) $(x, y) = (x', y')$, with a probability of $\frac{1}{N}$.

(b) $(x, y) \neq (x', y')$, with a probability of $\frac{N-1}{N}$.

The storage time to (x, y) should therefore be:

$$t_s(x,y) = \frac{1}{N} \times t_s^a + \frac{N-1}{N} \times t_s^b \qquad \text{(Eq. 4.1)}$$

where t_s^a is the operation time for case (a) and t_s^b for case (b).

Claim 4.1 *The operation time for case* (a) is:

$$t_s^a(x,y) = x + (y-1) \qquad \text{(Eq. 4.2)}$$

Proof: The preceding operation in this case must be retrieval so that the cell can be emptied for this storage operation. Therefore, according to Property 4.1, we have $d_v = 1$ and $d_h = 0$. With Property 4.2, the operation is then represented as:

$$\begin{aligned} t_s^a(x,y) &= max(y-1, 0) + max(0, x) \\ &= x + (y-1) \end{aligned}$$ ∎

For case (b), we should further decompose it into two sub-cases as follows:

(b1) $y = y'$, with a probability of $\frac{B-1}{N-1}$: the preceding operation is to the same tier but a different bay, which can be a storage or a retrieval with same probability, i.e., 0.5.

(b2) $y \neq y'$, with a probability of $\frac{N-B}{N-1}$: the preceding operation is to a different tier, which can be a storage or a retrieval with same probability, i.e., 0.5.

The storage time $t_s^b(x, y)$ is therefore represented as:

$$t_s^b(x,y) = \frac{B-1}{N-1} \times t_s^{b1} + \frac{N-B}{N-1} \times t_s^{b2} \qquad \text{(Eq. 4.3)}$$

where t_s^{b1} is the operation time for case (b1) and t_s^{b2} for case (b2).

Claim 4.2 *The operation time for case* (b1) is:

$$t_s^{b1}(x,y) = x + \frac{y-1}{2} + \frac{1}{2(B-1)} \sum_{(1 \leqslant x' \leqslant B) \bigwedge (x' \neq x)} max(2(y-1); x') \qquad \text{(Eq. 4.4)}$$

Proof: The preceding operation is either a storage or a retrieval with same probability 0.5. From Property 4.1 and 4.2, if the preceding operation is a storage, t_s^{b1} is:

$$t_s^{b1}(x,y) = \frac{1}{B-1} \sum_{(1 \leqslant x' \leqslant B) \bigwedge (x' \neq x)} max(2(y-1), x') + x$$

Else it is:

$$t_s^{b1}(x,y) = x + (y-1)$$

Combine the above two equation and get their weighted average with same weight 0.5, we have:

$$\begin{aligned} t_s^{b1}(x,y) &= \frac{1}{2}(x + (y-1)) + \frac{1}{2(B-1)} \sum_{(1 \leqslant x' \leqslant B) \bigwedge (x' \neq x)} max(2(y-1), x') + x \\ &= x + \frac{y-1}{2} + \frac{1}{2(B-1)} \sum_{(1 \leqslant x' \leqslant B) \bigwedge (x' \neq x)} max(2(y-1), x') \end{aligned}$$

∎

We further investigate the summation part in the last term of Eq. 4.4. If $2(y-1) < B$, it can be separated as:

$$\begin{aligned} &\sum_{(1 \leqslant x' \leqslant B) \bigwedge (x' \neq x)} max(2(y-1), x') \\ &= \sum_{x'=1}^{2(y-1)} 2(y-1) + \sum_{x'=2(y-1)+1}^{B} x' - max(2(y-1), x) \\ &= 4(y-1)^2 + \left(\frac{B(B+1)}{2} - \frac{(2y-2)(2y-1)}{2}\right) - max(2(y-1), x) \\ &= 2y^2 - 5y + \frac{B^2 + B + 6}{2} - max(2(y-1), x) \end{aligned}$$

Whereas if $2(y-1) \geqslant B$, it can be simplified as:

$$\begin{aligned} &\sum_{(1 \leqslant x' \leqslant B) \bigwedge (x' \neq x)} max(2(y-1), x') \\ &= \sum_{(1 \leqslant x' \leqslant B) \bigwedge (x' \neq x)} 2(y-1) \\ &= 2(B-1)(y-1) \end{aligned}$$

Therefore, Eq. 4.4 can be simplified as the close from function of x and y:

$$t_s^{b1}(x,y) = \begin{cases} \frac{1}{B-1}y^2 - \frac{B+6}{2(B-1)}y + \frac{B^2-B+8}{4(B-1)} + x - \frac{max(2(y-1),x)}{2(B-1)} & \text{if } 2(y-1) < B, \\ x + \frac{3}{2}y - \frac{3}{2} & \text{otherwise.} \end{cases} \quad \text{(Eq. 4.5)}$$

Claim 4.3 *The operation time for case* (b2) is:

$$t_s^{b2} = \frac{(B+1)(2T-3)}{4B(T-1)}y + x + \frac{(B+1)(T^2-5T+6)}{4B(T-1)} + \frac{1}{4B(T-1)} \sum_{(1\leqslant y'\leqslant T)\wedge(y'\neq y)} \sum_{(1\leqslant x''\leqslant B)\wedge(x''\neq x)} (max(y-1+y'-1,x'') + max(y-1,x'')) \quad \text{(Eq. 4.6)}$$

where x'' is the bay location of the previous operation on tier y.

The proof of this claim involves further decomposition of the case and therefore is given in Appendix A. Now we have the expressions of the operation times for all considered cases. In summary, we combine all the equations, with their probabilities as weights, to yield the equation for the storage time to cell (x, y) as follows:

$$t_s(x,y) = \frac{1}{N} \times t_s^a + \frac{B-1}{N} \times t_s^{b1} + \frac{N-B}{N} \times t_s^{b2} \quad \text{(Eq. 4.7)}$$

Substituting Eq. 4.2, Eq. 4.5, and Eq. 4.6 into the above Eq. 4.7, the final expression of the storage time could be obtained. Due to the complexity of the final equation, it is not displayed here. In application, it is more convenient to calculate Eq. 4.2, Eq. 4.5, and Eq. 4.6 individually, and substitute the results into Eq. 4.7, instead of working with the long expression.

Moreover, the overall average storage time of the rack can be straightforwardly derived as follows:

$$t_s = \frac{1}{T \cdot B} \sum_{1\leqslant x\leqslant B} \sum_{1\leqslant y\leqslant T} t_s(x,y) \quad \text{(Eq. 4.8)}$$

This is useful for estimating the storage performance for the preliminary evaluation of AS/RS design configurations.

4.2.2 Retrieval time model

The algorithm for calculating storage time is applicable to developing retrieval time model. The retrieval operation could be decomposed into two phases:

(1) The VP moves from tier d_v to tier y; at the same time, the HP at tier y moves from d_h to bay x to retrieve the load, and then moves to the handover station for the transfer of the load.

(2) After the transfer, the VP at tier y moves to I/O station to deliver the load.

Property 4.1 and 4.2 hold, and the possible cases are similar to those of a storage operation. Therefore the processes for deriving the retrieval time model are simply the same as the previous section. In particular, we have the following preparing claims for the model:

Claim 4.4 *The retrieval time* t_r^a *for case* (a) is the linear function of x and y:

$$t_r^a = x + y - 1 \qquad \text{(Eq. 4.9)}$$

Claim 4.5 *The retrieval time* t_r^{b1} *for case* (b1) is represented as:

$$t_r^{b1} = \frac{1}{2}x + y - 1 + \frac{1}{2}max(y-1, 2x) + \frac{1}{2(B-1)} \sum_{(1 \leqslant x' \leqslant B) \bigwedge (x' \neq x)} |x' - x| \qquad \text{(Eq. 4.10)}$$

Claim 4.6 *The retrieval time* t_r^{b2} *for case* (b2) is given as:

$$\begin{aligned} t_r^{b2} = {} & (y-1) + \frac{1}{2B}max(x, y-1) + \frac{B-1}{4B}max(2x, y-1) \\ & + \frac{1}{4B(T-1)} \sum_{(1 \leqslant y' \leqslant T) \bigwedge (y' \neq y)} 2max(x, |y'-y|) + (B-1)max(2x, |y'-y|) \\ & + \frac{1}{4B} \sum_{(1 \leqslant x'' \leqslant B) \bigwedge (x'' \neq x)} max(|x''-x| + x, y'-1) \\ & + \frac{1}{4B(T-1)} \sum_{(1 \leqslant y' \leqslant T) \bigwedge (y' \neq y)} \sum_{(1 \leqslant x'' \leqslant B) \bigwedge (x'' \neq x)} max(|x''-x| + x, |y'-y|) \end{aligned} \qquad \text{(Eq. 4.11)}$$

Claim 4.7 *The retrieval time to cell (x, y) can be represented as the combination of various cases with different probabilities:*

$$t_r(x,y) = \frac{1}{N} \times t_r^a + \frac{B-1}{N} \times t_r^{b1} + \frac{N-B}{N} \times t_r^{b2} \qquad \text{(Eq. 4.12)}$$

The proofs of these claims are given in Appendix A, which take similar processes as those for deriving the storage time model. The overall average retrieval time of the rack can be obtained by summing up the retrieval times from all the cells and taking average over the number of cells.

4.2.3 Verification of the discrete travel time model

In this section, we examine the derived operation time models and discuss their applications. Note that although the travel time model is done on a single rack with one VP, the result is also valid for a dual-rack structure with two VPs and two I/O stations as long as the sequence of access operations are carefully planned such that the two VPs do not compete for the shared HPs at the same time.

The evaluations of the models are done by comparing them with computer simulation results. The algorithm for the simulations is summarized in next page in "Simulation Algorithm". 100,000 jobs (which is considerably large compared with the number of cells in an AS/RS rack) were executed in each experiment to simulate the infinite sequence of operations, where $\alpha = 0.5$. The initial status of the rack was set to unknown, which means that if a cell was selected for the first time to do an operation, it could be executed regardless of its operation type.

SIMULATION ALGORITHM :

(1) initialize the rack and the percentage of storage operations, i.e., α.

(2) generate a sequence of job with an attribute indicating the operation type.

(3) execute the jobs by randomly selecting:

  (a) an open location for a storage job.

  (b) a full location for a retrieval job.

(4) terminate the simulation when:

  (a) no open location can be found for a storage job.

  (b) no full location can be found for a retrieval job.

  (c) all jobs in the sequence are done.

In order to test the sensitivity of the models to the rack configurations, the experiments were done based on two different racks, one with 144 cells and the other with 288 cells. Table 4.1 gives the details of the two rack layouts. 5 replications of simulation with different random seeds were executed for each layout, which were able to achieve a relative error of 0.01 at 95% confidence level. The outputs of the models and the simulation results are summarized in Table 4.2 and 4.3.

Note: % Dev refers to % deviation which was used to indicate the accuracy of the models in the current research and calculated as:

$$\%Dev = \frac{modelresults - simulationresults}{simulationresults} \times 100\%$$

The results show that for sufficiently long sequences of accesses, the discrete models can accurately describe the operation times for various rack configurations. The average error is at an acceptable level of 0.14% with the maximum error not exceeding 0.4%, unsensitive to the discreteness of the rack.

The travel time models are useful for the AS/RS rack configuration design. For a desired throughput requirement, we can obtain a set of possible rack configurations that yield a

Configuration	Layout 1		Shape factor	Layout 2		Shape factor
	Tiers	Bays		Tiers	Bays	
1	3	48	0.125	4	72	0.111
2	6	24	0.500	8	36	0.444
3	9	16	1.125	12	24	1.000
4	12	12	2.000	16	18	1.778
5	–	–	–	18	16	2.250
6	16	9	3.556	24	12	4.000
7	24	6	8.000	36	8	9.000
8	48	3	32.000	72	4	36.000

Table 4.1: Details of different rack configurations

Configuration	Simulation results	Model results	Errors (% Dev)
1	95.023	94.980	0.046
2	56.845	56.966	0.212
3	50.944	51.050	0.208
4	54.447	54.482	0.064
5	–	–	–
6	64.524	64.752	0.354
7	91.341	91.246	0.105
8	178.929	178.639	0.162

Table 4.2: Comparisons of discrete models with simulation results (Layout 1)

Configuration	Simulation results	Model results	Errors (% Dev)
1	141.912	141.975	0.044
2	83.483	83.723	0.288
3	73.001	73.085	0.116
4	75.994	76.202	0.273
5	80.174	80.203	0.036
6	96.935	96.865	0.072
7	137.421	137.527	0.078
8	269.564	269.303	0.097

Table 4.3: Comparisons of discrete models with simulation results (Layout 2)

corresponding expected operation time from the models. This is simplified as looking for a line from a surface as shown in Figure 4.3. Together with other constraints, such as shape factor, we can preliminarily decide an optimal rack configuration. On the other hand, for a particular rack configuration, the result suggests a class-based storage assignment policy. For instance, Figure 4.4 illustrates possible class boundaries of zones based on the operation times for a 12-by-28 rack.

We can also use the model to estimate the optimal shape factor b, which represents the shape of storage rack in terms of time, namely:

$$b = \frac{\text{travel time to the highest tier}}{\text{travel time to the farthest bay}}$$

An experiment was done which varies b from $[0.1, 4]$ in five rack configurations to investigate its impact on the operation time and to find out an optimal b. The results are illustrated in Figure 4.5.

In this section, we have demonstrated that the discrete travel time model is computationally feasible. However, it is obvious that the expressions are rather complicated and to some extent not practical for quick estimation of operation time for an AS/RS rack.

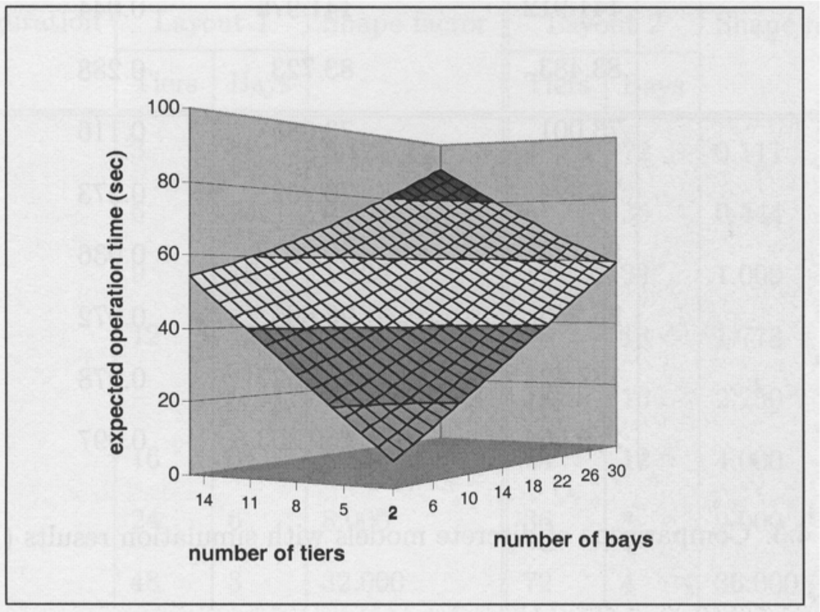

Figure 4.3: Variation of operation time

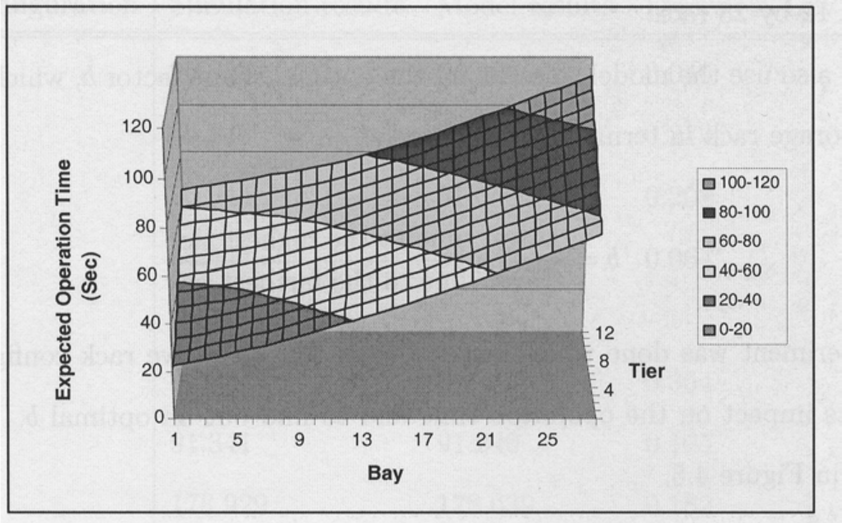

Figure 4.4: Zoning of rack by operation time

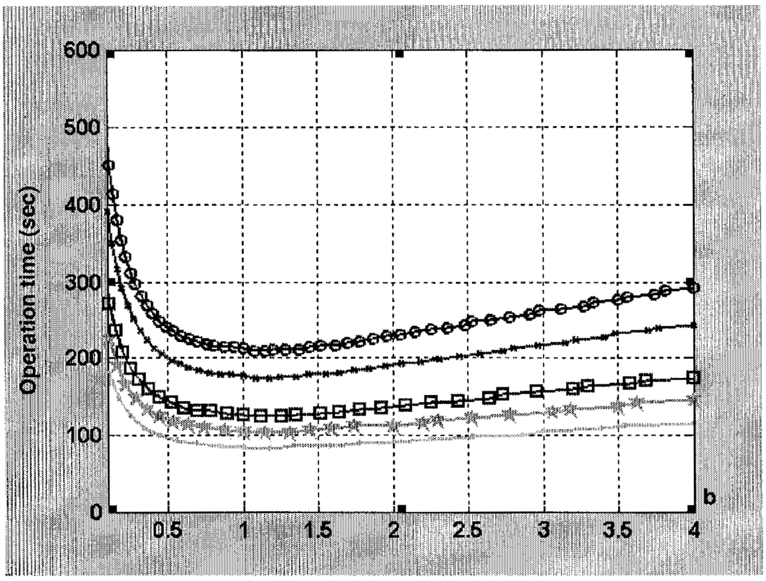

Figure 4.5: Influence of shape factor on operation time

Inspired by the literature that utilizes continuous models to simplify the problem, in the next section we aim at building the continuous expressions for the SP-AS/RS.

4.3 Continuous time models

If the discrete rack is regarded as a continuous pick face, we can deploy integration technique to reduce the difficulty of the subsequent analysis and simplify the travel time expressions [Boz84]. Moreover, we introduce a parameter α to represent the ratio of storage operations in the sequence, which relaxes the infinite sequence assumption for the discrete model and therefore generalizes the derived continuous models. In summary, the assumptions are made as follows throughout this section:

(1) The rack is considered to be a continuous rectangular pick face.

(2) Randomized storage is used, which means that any point within the pick face is equally likely to be selected for storage or retrieval.

(3) The system operates on single command basis for unit loads.

(4) Dwell point policy for platforms is that they will stay where they are at the end of each operation.

(5) Each operation in the sequence of operations is independent of the previous operations.

(6) Pick-up, deposit, and load transfer times associated with load handling are ignored.

(7) Specifications of the rack and the platforms are known. The platforms travel at constant speeds.

(8) There is no concurrent movement of platforms for different operations.

We introduce the following notations to normalize the rectangular pick face with length of 1 and height of b in terms of time:

- R_L: the length of the rack.
- R_H: the height of the rack.
- v_h: the speed of the HPs.
- v_v: the speed of the VPs.
- t_h: the travel time required for an HP to go to the farthest bay, i.e., $t_h = \frac{R_L}{v_h}$.
- t_v: the travel time required for a VP to go to the highest tier, i.e., $t_v = \frac{R_H}{v_v}$.
- b: shape factor representing the shape of the rack in terms of time, $b = \frac{t_v}{t_h}$.

Note that the vertical and horizontal movements in the proposed SP-AS/RS are not symmetric, we should assume b to an arbitrary positive value instead of $0 < b \leqslant 1$ in the literature.

It is worthwhile to investigate the relationship between the current operation and its previous operations. We know that the expected travel time of an operation is dependent on the starting positions of the VP and HP to perform this operation, which are themselves decided by the types of previous operations. The starting position of the VP, say d_v, is decided by the immediately preceding operation, i.e., $d_v = y_1$ if it was a storage and $d_v = 1$ if it was a retrieval. The starting position of the associated HP, d_h, is decided by its previous operation, i.e., $d_h = 0$ if it was a storage and $d_h = x_1$ if it was a retrieval. In summary, three operations are involved in determining the travel time for an operation:

I Current operation (CO): determines the destination of the operation, denoted as (x_2, y_2).

II Preceding operation (PO): determine the starting position of VP, denoted as y_1.

III Last operation on the associated HP (LO): determine the starting position of HP, denoted as (x_1, y_2).

Note that all these coordinates are measured in time. For randomized storage, the coordinate locations are assumed to be uniformly distributed. Thus the probability distribution function (PDF) and the probability density function (pdf) of $x_i \quad (i = 1, 2)$ are:

$$F_{X_i}(v) = \begin{cases} v & 0 \leqslant v \leqslant 1 \\ 1 & v \geqslant 1 \end{cases} \quad \text{and} \quad f_{X_i}(v) = \begin{cases} 1 & 0 \leqslant v \leqslant 1 \\ 0 & v \geqslant 1 \end{cases} \qquad \text{(Eq. 4.13)}$$

Similarly, the PDF and pdf of $y_i \quad (i = 1, 2)$ can be represented as:

$$F_{Y_i}(y_i \leqslant v) = \begin{cases} \frac{v}{b} & 0 \leqslant v \leqslant b \\ 1 & v \geqslant b \end{cases} \quad \text{and} \quad f_{Y_i}(v) = \begin{cases} \frac{1}{b} & 0 \leqslant v \leqslant b \\ 0 & v \geqslant b \end{cases} \qquad \text{(Eq. 4.14)}$$

d on the above analysis and statistical arguments, we can conclude that:

Property 4.3 *The expected travel time for an operation is the average of the operation times under all the permutations of the preceding operation and the previous operation at the target tier.*

Defining an operation as a storage or a retrieval, the expected travel time per operation E_{t_o} can be shown to be:

$$E(t_o) = P_s E(t_s) + P_r E(t_r) \qquad \text{(Eq. 4.15)}$$

where $E(t_s)$ and $E(t_r)$ denote the expected storage time and the expected retrieval time, respectively, while P_s and P_r indicate their probabilities. Now we consider $E(t_s)$ and $E(t_r)$ separately.

4.3.1 Expected storage time

There is α percentage of the operations are storages and that three operations are associated with the expected travel time of an operation, i.e., CO, PO, and LO. Now that CO is a storage, from Property 4.3, we should consider the following permutations of PO and LO:

(1) PO and LO are both storage, whose probability is $P_1 = \alpha^2$.

(2) PO is storage and LO is retrieval, whose probability is $P_2 = \alpha(1-\alpha)$.

(3) PO is retrieval and LO is storage, whose probability is $P_3 = \alpha(1-\alpha)$.

(4) PO and LO are both retrieval, whose probability is $P_4 = (1-\alpha)^2$.

Subsequently, we formulate the travel time for a storage as follows:

$$t_s = P_1 \cdot t_1 + P_2 \cdot t_2 + P_3 \cdot t_3 + P_4 \cdot t_4$$

whereas the expected storage time is represented as:

$$E(t_s) = P_1 \cdot E(t_1) + P_2 \cdot E(t_2) + P_3 \cdot E(t_3) + P_4 \cdot E(t_4) \qquad \text{(Eq. 4.16)}$$

Solving the components individually will yield the model for expected storage time.

Claim 4.8 *The storage time if PO and LO are both storage can be calculated as:*

$$t_1 = max(y_1 + y_2, x_1) + x_2,$$

whose expected value is:

$$E(t_1) = \begin{cases} \frac{7}{12}b^2 + \frac{1}{2} & 0 < b \leqslant \frac{1}{2} \\ -\frac{1}{12}b^2 + \frac{4}{3}b + \frac{1}{3b} - \frac{1}{24b^2} - \frac{1}{2} & \frac{1}{2} < b \leqslant 1 \\ b + \frac{1}{24b^2} & b > 1 \end{cases} \qquad \text{(Eq. 4.17)}$$

The proof is given in Appendix I.

Claim 4.9 *The storage time if PO is storage and LO is retrieval can be calculated as:*

$$t_2 = y_1 + y_2 + x_2,$$

whose expected value is:

$$E(t_2) = b + \frac{1}{2} \qquad \text{(Eq. 4.18)}$$

Proof: In this case, VP starts at y_1 and the associated HP starts at handover station. According to Property 4.2, $t_2 = max(y_1 + y_2, 0) + x_2 = y_1 + y_2 + x_1$. As for the expected value, we have the pdf of x and y in Eq. 4.13 and Eq. 4.14, therefore with the independence assumption on x and y,

$$\begin{aligned} E(t_2) &= E(y_1) + E(y_2) + E(x_1) \\ &= 2\int_0^b y \cdot f_Y(y)dy + \int_0^1 x \cdot f_X(x)dx \\ &= b + \frac{1}{2} \end{aligned}$$

∎

Claim 4.10 *The storage time if PO is retrieval and LO is storage can be calculated as:*

$$t_3 = max(y_2, x_1) + x_2,$$

whose expected value is:

$$E(t_3) = \begin{cases} \frac{1}{6}b^2 + 1 & 0 < b \leqslant 1 \\ \frac{b}{2} + \frac{1}{6b} + \frac{1}{2} & b \geqslant 1 \end{cases} \qquad \text{(Eq. 4.19)}$$

Proof: In this case, VP starts at I/O station and the associated HP starts at (x_1, y_2). According to Property 4.2, $t_3 = max(y_2, x_1) + x_2$. As for the expected value, we have the PDF of x and y in Eq. 4.13 and Eq. 4.14, therefore with the independence assumption on x and y:

$$\begin{aligned} F_Z(z) &= F_Z(max(y_2, x_1) \leqslant z) \\ &= F_Z(y_2 \leqslant z)F_Z(x_1 \leqslant z) \\ &= \begin{cases} \frac{z^2}{b} & 0 \leqslant z \leqslant b \quad \text{and} \quad 0 < b \leqslant 1 \\ z & z > b \quad \text{and} \quad 0 < b \leqslant 1 \\ \frac{z^2}{b} & 0 \leqslant z \leqslant 1 \quad \text{and} \quad b \geqslant 1 \\ \frac{z}{b} & 1 \leqslant z \leqslant b \quad \text{and} \quad b \geqslant 1 \\ z & z \geqslant b \quad \text{and} \quad b \geqslant 1 \end{cases} \end{aligned}$$

Hence:

$$f_Z(z) = \begin{cases} \frac{2z}{b} & 0 \leqslant z \leqslant b \quad \text{and} \quad 0 < b \leqslant 1 \\ 1 & z > b \quad \text{and} \quad 0 < b \leqslant 1 \\ \frac{2z}{b} & 0 \leqslant z \leqslant 1 \quad \text{and} \quad b \geqslant 1 \\ \frac{1}{b} & 1 \leqslant z \leqslant b \quad \text{and} \quad b \geqslant 1 \\ 1 & z \geqslant b \quad \text{and} \quad b \geqslant 1 \end{cases}$$

Now we can get the expected value of t_3 as:

$$\begin{aligned} E(t_3) &= E(max(y_2, x_1)) + E(x_2) \\ &= \int_0^\infty z \cdot f_Z(z)dz + \int_0^1 x \cdot f_X(x)dx \\ &= \begin{cases} \frac{b^2}{6} + 1 & 0 < b \leqslant 1 \\ \frac{b}{2} + \frac{1}{6b} + \frac{1}{2} & b \geqslant 1 \end{cases} \end{aligned}$$

∎

Claim 4.11 *The storage time if PO and LO are both retrieval can be calculated as:*

$$t_4 = y_2 + x_2,$$

whose expected value is:

$$E(t_4) = \frac{b}{2} + \frac{1}{2} \qquad \text{(Eq. 4.20)}$$

Proof: In this case, VP starts at I/O station and the associated HP starts at handover station. According to Property 4.2, $t_2 = max(y_2, 0) + x_2 = y_2 + x_2$. As for the expected value, we have the pdf of x and y in Eq. 4.13 and Eq. 4.14, therefore with the independence assumption on x and y,

$$\begin{aligned} E(t_4) &= E(y_2) + E(x_2) \\ &= \int_0^b y \cdot f_Y(y)dy + \int_0^1 x \cdot f_X(x)dx \\ &= \frac{b}{2} + \frac{1}{2} \end{aligned}$$

∎

Substituting Eq. 4.17 to Eq. 4.20 into Eq. 4.16, we obtain the expected storage time as:

$$E(t_s) = \begin{cases} \dfrac{5\alpha^2 + 2\alpha}{12} b^2 + \dfrac{1-\alpha^2}{2} b + \dfrac{1+\alpha}{2} & 0 < b \leqslant \dfrac{1}{2} \\ \dfrac{-3\alpha^2 + 2\alpha}{12} b^2 + \dfrac{5\alpha^2 + 3}{6} b + \dfrac{\alpha^2}{3b} - \dfrac{\alpha^2}{24b^2} + \dfrac{-2\alpha^2 + \alpha + 1}{2} & \dfrac{1}{2} \leqslant b \leqslant 1 \\ \dfrac{\alpha + 1}{2} b + \dfrac{-\alpha^2 + \alpha}{6b} + \dfrac{\alpha^2}{24b^2} + \dfrac{1}{2} & b \geqslant 1 \end{cases} \tag{Eq. 4.21}$$

4.3.2 Expected retrieval time

For retrieval time model, similarly we should consider the four permutations of PO and LO and solve them individually to get their weighted average as the expected retrieval time. However, note that the VP or HP could go directly from their dwell positions to perform a retrieval operation, the calculation of travel times may involve taking the absolute value of the platform displacements. Therefore the processes are much more complicated and hereby we only give the results, leaving the proofs to Appendix II.

Recall that the four permutations of PO and LO are PO/storage with LO storage, PO/storage with LO/retrieval, PO/retrieval with LO storage, and PO/retrieval with LO/retrieval. The retrieval times under these circumstances are given as follows:

Claim 4.12 *The retrieval time if PO and LO are both storage can be represented as:*

$$E(t_1) = \begin{cases} \dfrac{b^3}{40} + \dfrac{b}{2} + \dfrac{5}{6} & 0 < b \leqslant 1 \\ \dfrac{-b^3}{120} + \dfrac{b^2}{12} + \dfrac{b}{2} + \dfrac{1}{6b} - \dfrac{1}{20b^2} + \dfrac{2}{3} & 1 \leqslant b \leqslant 2 \\ \dfrac{5b}{6} + \dfrac{5}{6b} - \dfrac{19}{60b^2} & b \geqslant 2 \end{cases} \tag{Eq. 4.22}$$

Claim 4.13 *The retrieval time if PO is storage and LO is retrieval can be represented as:*

$$E(t_1) = \begin{cases} \dfrac{b^2}{24} + \dfrac{b}{2} + 1 & 0 < b \leqslant 2 \\ \dfrac{5b}{6} + \dfrac{4}{3b} - \dfrac{2}{3b^2} & b \geqslant 2 \end{cases} \tag{Eq. 4.23}$$

Claim 4.14 *The retrieval time if PO is retrieval and LO is storage can be represented as:*

$$E(t_1) = \begin{cases} \dfrac{b^3}{16} + \dfrac{5}{6} & 0 < b \leqslant 1 \\ \dfrac{-b^3}{48} + \dfrac{b^2}{6} + \dfrac{1}{12b} + \dfrac{2}{3} & 1 \leqslant b \leqslant 2 \\ \dfrac{b}{2} + \dfrac{5}{12b} & b \geqslant 2 \end{cases} \tag{Eq. 4.24}$$

Claim 4.15 *The retrieval time if PO and LO are both retrieval can be represented as:*

$$E(t_1) = \begin{cases} \dfrac{b^2}{12} + \dfrac{b}{2} + 1 & 0 < b \leqslant 2 \\ b + \dfrac{2}{3b} & b \geqslant 2 \end{cases} \tag{Eq. 4.25}$$

Recall that the expected retrieval time is the combination of the above four cases:

$$\begin{aligned} E(t_r) &= P_1 \cdot E(t_1) + P_2 \cdot E(t_2) + P_3 \cdot E(t_3) + P_4 \cdot E(t_4) \\ &= \alpha^2 \cdot E(t_1) + (1-\alpha)\alpha \cdot E(t_2) + \alpha(1-\alpha) \cdot E(t_3) + (1-\alpha)^2 \cdot E(t_4) \end{aligned}$$

Substituting Eq. 4.22 to Eq. 4.25 into the above equation yields the expected retrieval time model:

$$E(t_r) = \begin{cases} \dfrac{-3\alpha^2+5\alpha}{80}b^3 + \dfrac{\alpha^2-3\alpha+2}{24}b^2 + \dfrac{b}{2} + \dfrac{6-\alpha}{6} & 0 < b \leqslant 1 \\ \dfrac{3\alpha^2+5\alpha}{240}b^3 + \dfrac{1\alpha^2+\alpha+2}{24}b^2 + \dfrac{b}{2} + \dfrac{\alpha^2+\alpha}{12b} - \dfrac{\alpha^2}{20b^2} + \dfrac{3-\alpha}{3} & 1 \leqslant b \leqslant 2 \\ \dfrac{6-\alpha}{6}b + \dfrac{-3\alpha^2+5\alpha+8}{12b} + \dfrac{21\alpha^2-40\alpha}{60b^2} & b \geqslant 2 \end{cases} \qquad \text{(Eq. 4.26)}$$

Now we are ready to calculate the expected operation time, which is the combination of the expected storage and retrieval times. Since P_s for probability of storage operation is α and P_r for that of retrieval is $1-\alpha$, based on Eq. 4.16, Eq. 4.26 and Eq. 4.15, we have:

$$E(t_o) = \begin{cases} \frac{3\alpha^3-8\alpha^2+5\alpha}{80}b^3 + \frac{9\alpha^3+8\alpha^2-5\alpha+2}{24}b^2 + \frac{1-\alpha^3}{2}b + \frac{3-2\alpha+2\alpha^2}{3} & 0 < b \leqslant \frac{1}{2} \\ \frac{3\alpha^3-8\alpha^2+5\alpha}{80}b^3 + \frac{-7\alpha^3+8\alpha^2-5\alpha+2}{24}b^2 + \frac{3+5\alpha^3}{6}b + \frac{\alpha^3}{3b} - \frac{\alpha^3}{24b^2} + \frac{3-2\alpha+2\alpha^2-3\alpha^3}{3} & \frac{1}{2} \leqslant b \leqslant 1 \\ \frac{-3\alpha^3+8\alpha^2-5\alpha}{240}b^3 + \frac{\alpha^3-2\alpha^2-\alpha+2}{24}b^2 + \frac{1+\alpha^2}{2}b + \frac{-3\alpha^3+2\alpha^2+\alpha}{12b} + \frac{11\alpha^3-6\alpha^2}{120b^2} + \frac{2\alpha^2-5\alpha+6}{6} & 1 \leqslant b \leqslant 2 \\ \frac{\alpha^2-2\alpha+3}{3}b + \frac{\alpha^3-6\alpha^2-3\alpha+8}{12b} + \frac{-37\alpha^3+122\alpha^2-80\alpha}{120b^2} + \frac{\alpha}{2} & b \geqslant 2 \end{cases} \qquad \text{(Eq. 4.27)}$$

Observations: Considering an infinite sequence of operations, the value of α should be $\dfrac{1}{2}$, and the continuous model becomes comparable to the discrete model. We hereby give the continuous expression of the travel time when $\alpha = \dfrac{1}{2}$ for comparison purpose:

$$E(t_o)\bigg|_{\alpha=\frac{1}{2}} = \begin{cases} \dfrac{7}{160}b^3 + \dfrac{7}{64}b^2 + \dfrac{7}{16}b + \dfrac{5}{6} & 0 < b \leqslant \dfrac{1}{2} \\ \dfrac{7}{160}b^3 + \dfrac{5}{192}b^2 + \dfrac{29}{48}b + \dfrac{1}{24b} - \dfrac{1}{192b^2} + \dfrac{17}{24} & \dfrac{1}{2} \leqslant b \leqslant 1 \\ \dfrac{-7}{1920}b^3 + \dfrac{3}{64}b^2 + \dfrac{5}{8}b + \dfrac{5}{96b} - \dfrac{1}{960b^2} + \dfrac{2}{3} & 1 \leqslant b \leqslant 2 \\ \dfrac{5}{6}b + \dfrac{41}{96b} - \dfrac{113}{960b^2} + 1 & b \geqslant 2 \end{cases} \qquad \text{(Eq. 4.28)}$$

Configuration	Shape factor	Simulation results	Model results	Errors (% Dev)
1	0.125	95.023	96.093	1.126
2	0.500	56.845	58.363	2.670
3	1.125	50.944	52.899	3.836
4	2.000	54.447	56.721	4.176
5	NA	NA	NA	NA
6	3.556	64.524	67.306	4.312
7	8.000	91.341	94.071	2.989
8	32.000	178.929	181.777	1.591

Table 4.4: Comparisons of continuous models with simulation results (Layout 1)

4.3.3 Verification of the continuous travel time model

We conducted same experiments as described in Section 4.2.3 to verify the continuous models and compare them with the discrete models. Recall that we designed two racks with 144 storage cells and 288 storage cells, respectively, to test the impact of rack discreteness. Intuitively due to the continuous approximation, we assume that rack discreteness would affect the accuracy of the continuous models and therefore we purposely design a rack with same area but with extremely large number of cells (2592 in layout 3). These results of $\alpha = 0.5$ are shown in Tables 4.4 to 4.6.

Note that there are 10 configurations for Layout 3, the shape factors of which are not all same as those of the first two layouts. The results show that the continuous models perform satisfactorily with a maximum error of around 4%. From the above tables, it can also be observed that the accuracy of the models increases with the number of storage cells in the rack. This affirms that the discreteness of the rack has impact on the continuous models. However, in practice the size of the cells would be fixed, e.g., $4.5m \times 4.5m$ for container (considerably large items) storage in our case, while the area of the rack could be more flexibly decided. Therefore, if we pay attention to the rack discreteness when applying the continuous model, its accuracy can be assured at an acceptable level.

Configuration	Shape factor	Simulation results	Model results	Errors (% Dev)
1	0.111	141.912	143.096	0.834
2	0.444	83.483	85.078	1.911
3	1.000	73.001	74.841	2.521
4	1.778	75.994	78.344	3.092
5	2.250	80.174	82.496	2.896
6	4.000	96.935	99.434	2.578
7	9.000	137.421	140.328	2.116
8	36.000	269.564	272.356	1.036

Table 4.5: Comparisons of continuous models with simulation results (Layout 2)

Configuration	Shape factor	Simulation results	Model results	Errors (% Dev)
1	0.063	184.862	185.999	0.615
2	0.444	84.455	85.078	0.737
3	1.000	74.198	74.841	0.866
4	1.778	77.551	78.344	1.021
5	2.250	81.658	82.496	1.027
6	4.000	98.549	99.434	0.898
7	9.000	139.156	140.328	0.842
8	36.000	271.131	272.356	0.452
9	64.000	360.410	361.732	0.367
10	576.000	1077.282	1080.564	0.305

Table 4.6: Comparisons of continuous models with simulation results (Layout 3)

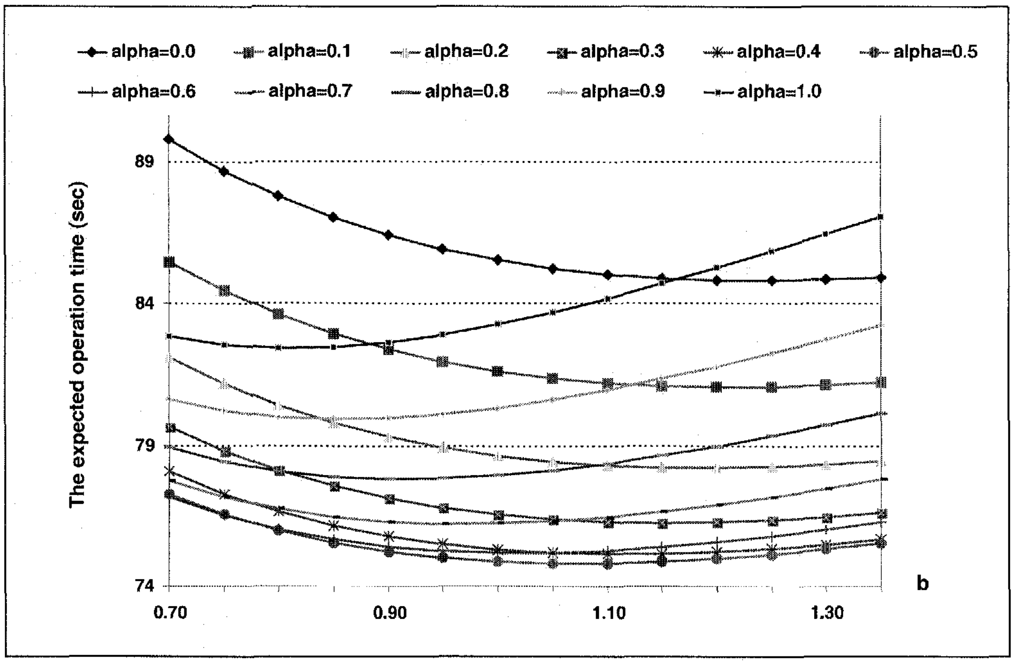

Figure 4.6: Sensitivity study of b

We then further investigate the impacts of shape factor b and the fraction of storages α on the expected operation time. Based on Eq. 4.27, we vary b in the interval of $[0.1, 5]$ and α in $[0, 1]$, and calculate the corresponding expected operation times. Obvious tendencies could be observed from the results, parts of which are illustrated in Figures 4.6 and 4.7.

From Figure 4.6, it can be observed that optimal shape factor b varies with the fractions of storages, α, but sits in the range of $[0.8, 1.25]$. On the other hand, the optimal α also varies with the values of b but within an interval of $[0.4, 0.6]$ as shown in Figure 4.7. The global optimal combination of b and α appears to be $(1.05, 0.5)$. This finding is consistent with that from the discrete model.

4.4 An alternative dwell point policy

In this section we consider another dwell point policy for the platforms following the completion of a storage or a retrieval operation. Under this policy, the VP returns to the I/O

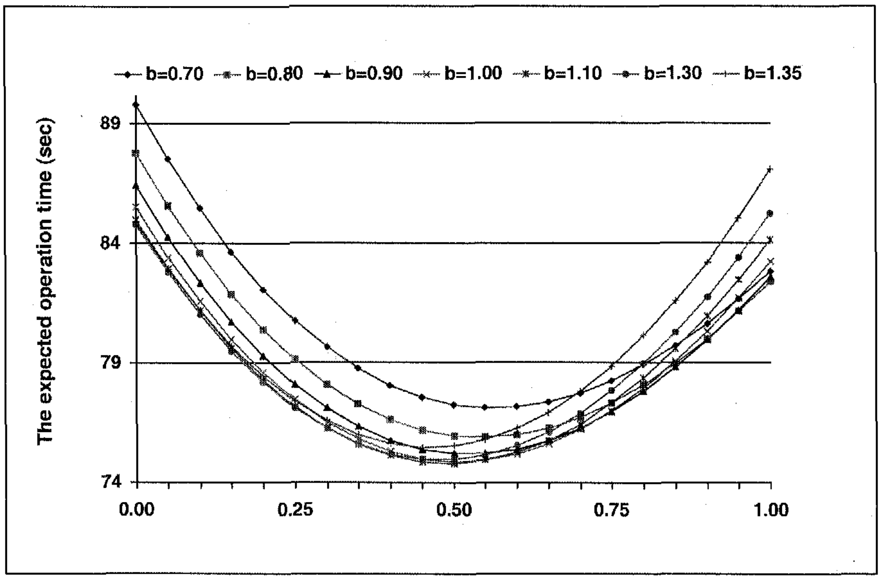

Figure 4.7: Sensitivity study of α

station and the HP returns to the handover station after finishing a job. We will build the discrete and continuous time models and compare its efficiency with the previous policy in terms of expected operation time. In this case, the dwell point of VP is $d_v = 1$, and that of HP is $d_v = 0$. Therefore, from Property 4.2, both of the storage time and retrieval time can be represented by:

$$t_s = t_r = y + max(y, 2x)$$

where (x, y) denotes the destination of the current operation.

Hence, regardless of the percentage of storage operations, the operation time to cell (x, y) is:

$$t_o(x, y) = y + max(y, 2x)$$

The expected operation time of a discrete rack can henceforth calculated as:

$$E(t_o)_{dis} = \frac{1}{T \cdot B} \sum_{y=2}^{T} \sum_{x=1}^{B} (y + max(y, 2x)) \qquad \text{(Eq. 4.29)}$$

Note that the VP is needed only for tier 2 and onwards, therefore in the summation y value starts from 2. The discrete expression is the accurate calculation of expected operation time.

Now we consider the continuous expression of the expected operation time. Let it be denoted by $E(t_o)_{con}$, we have:

$$E(t_o)_{con} = E(y + max(y, 2x)) = E(y) + E(max(y, 2x)) = E(y) + E(z) \qquad \text{(Eq. 4.30)}$$

where $z = max(y, 2x)$. Note that x and y are independent, the PDF of z can be expressed as:

$$F_Z(z) = P_r(Z \leqslant z) = P_r(Y \leqslant z, 2X \leqslant z) = P_r(2X \leqslant z)P_r(Y \leqslant z)$$

Recall that x and y follow the uniform distribution, whose pdf were given in Eq. 4.13 and Eq. 4.14. Therefore we have:

$$P_r(2X \leqslant z) = \begin{cases} 0 & z \leqslant 0 \\ \frac{z}{2} & 0 \leqslant z \leqslant 2 \\ 1 & z \geqslant 2 \end{cases} \quad \text{and} \quad P_r(Y \leqslant z) = \begin{cases} 0 & z \leqslant 0 \\ \frac{z}{b} & 0 \leqslant z \leqslant b \\ 1 & z \geqslant b \end{cases}$$

The PDF and pdf of z should be discussed in two cases. Case a: $0 \leqslant b \leqslant 2$

$$F_Z(z) = \begin{cases} 0 & z \leqslant 0 \\ \frac{z^2}{2b} & 0 \leqslant z \leqslant b \\ \frac{z}{2} & b \leqslant z \leqslant 2 \\ 1 & z \geqslant 2 \end{cases} \quad \text{and} \quad f_Z(z) = \begin{cases} \frac{z}{b} & 0 \leqslant z \leqslant b \\ \frac{1}{2} & b \leqslant z \leqslant 2 \\ 0 & \text{otherwise} \end{cases}$$

Case b: $b \geqslant 2$

$$F_Z(z) = \begin{cases} 0 & z \leqslant 0 \\ \dfrac{z^2}{2b} & 0 \leqslant z \leqslant 2 \\ \dfrac{z}{b} & 2 \leqslant z \leqslant b \\ 1 & z \geqslant 2 \end{cases} \quad \text{and} \quad f_Z(z) = \begin{cases} \dfrac{z}{b} & 0 \leqslant z \leqslant 2 \\ \dfrac{1}{b} & 2 \leqslant z \leqslant b \\ 0 & \text{otherwise} \end{cases}$$

Now we can derive the expected value of z as:

$$E(z) = \int_z z \cdot f_Z(z)dz = \begin{cases} \dfrac{b^2}{12} + 1 & 0 < b \leqslant 2 \\ \dfrac{b}{2} + \dfrac{2}{3b} & b \geqslant 2 \end{cases}$$

Recall that the expected value of y is $E(y) = \dfrac{b}{2}$. Therefore, substituting $E(y)$ and $E(z)$ into Eq. 4.30, we obtain the expected operation time as follows:

$$E(t_o)_{con} = \begin{cases} \dfrac{b^2}{12} + \dfrac{b}{2} + 1 & 0 < b \leqslant 2 \\ b + \dfrac{2}{3b} & b \geqslant 2 \end{cases} \qquad \text{(Eq. 4.31)}$$

Verification

For evaluation of the models, we use the previous Layout 2 with 288 cells and 8 configurations that vary the value of shape factor b from 0.11 to 36.0 (see Table 4.1 for details). The results are listed in Table 4.7.

As can be observed from the table, the maximum model error is about 3% which suggests that the continuous model satisfactorily describes the operation time under this dwell point policy. This assures that the continuous model works with a predictable performance.

Sensitivity analysis on b

It is worthwhile to investigate the influence of b on the expected operation time for an optimal rack design. Note that the expected operation time could be de-normalized as

Configuration	Shape factor	Discrete model	Continuous model	Errors (% Dev)
1	0.111	174.979	175.505	0.298
2	0.444	99.875	100.333	0.459
3	1.000	84.438	85.500	1.258
4	1.778	85.250	87.167	2.248
5	2.250	89.250	91.667	2.708
6	4.000	109.188	112.500	3.034
7	9.000	159.375	163.333	2.484
8	36.000	319.813	324.167	1.362

Table 4.7: Comparisons of continuous model with discrete model: dwell point policy 2

$T_{dnorm} = t_h \cdot E(t_o)$. Consider a fixed storage area A of the rack (in terms of time), we have $A = R_H \times R_L/v_h \times v_v = b \cdot t_h^2$. Therefore, substituting the above equation into the expression for T_{dnorm} and taking derivative with respect to b, we can find the local extreme:

(a): $0 < b \leqslant 2$

$$T_{dnorm} = \sqrt{\frac{A}{b}}(1 + \frac{b}{2} + \frac{b^2}{12})$$

therefore:

$$\frac{dT_{dnorm}}{db} = \frac{\sqrt{A}}{8}b^{-\frac{3}{2}}(b^2 + 2b - 4)$$

Setting the above equation to 0 and solving for b yields the value of b that leads to local extrema of T_{dnorm}, which is $b = 1.236$.

On the other hand, the second derivative of b:

$$\frac{d^2T_{dnorm}}{db^2} = \frac{\sqrt{A}}{16}b^{-\frac{5}{2}}((b-1)^2 + 11) > 0$$

Thus T_{dnorm} has a local minimum when $b = 1.236$. For the above rack layout in the experiment, the local minimum of T_{dnorm} is $T_{min} = 84.773$.

(b): $b \geqslant 2$

$$T_{dnorm} = \sqrt{\frac{A}{b}}(b + \frac{2}{3b})$$

therefore:

$$\frac{dT_{dnorm}}{db} = \frac{\sqrt{A}}{2} b^{-\frac{5}{2}} (b^2 - 2) > 0$$

which means that T_{dnorm} in this case increases with b and therefore has a local minimum when $b = 2$.

From the analyses of the above two cases, it is found that $b = 1.236$ is the global optimal for the rack layout under this dwell point policy.

Comparison of the two dwell point policies

In order to make the two dwell point policies comparable, we purposely used the same rack layout with 288 storage cells for experiments. In dwell policy 1, we found that the expected operation time had a minimum value of $E(t_o)^1_{min} = 74.781$seconds when $b = 1.05$ and $\alpha = 0.5$. In dwell point policy 2, the minimum value of expected operation time was found to be $E(t_o)^2_{min} = 84.773$seconds when $b = 1.236$. It can be concluded that policy 1 outperforms policy 2 for a balanced system where the fractions of storage and retrieval operations are identical. However, the expected operation time varies with α under policy 1, in some instances it exceeds 84.773 (as shown in Figure 4.7). Therefore, the optimal rack design will largely depend on the characteristics of the demand in our application. From a long-term point of view, the container yard has balanced work-flow and suggests that policy 1 be more preferable.

4.5 Operation Scheduling

In conventional AS/RS, the operation scheduling issue is mostly addressed by interleaving storage and retrieval operations to maximize the S/R machine utilization for improving throughput performance. However, interleaving rules require changes of the order in which the incoming requests arrive, which is unacceptable in container operations because the re-ordering of job sequences will complicate the traffic system in a container terminal. In other words, in the context of port operations, the storage yard should execute the operations in a predefined order. The additional constraint should be taken into account when doing operation scheduling for the SP-AS/RS.

4.5.1 Preprocessing of operations

In order not to change the sequencing of jobs, instead of interleaving storages and retrievals, we proposed a *preprocessing scheme* to pre-fetch containers for incoming retrieval requests (or pre-positioning the HPs for storages) to the handover stations to achieve the reduction in the cycle time of an operation. In the current design, on each tier of an SP-AS/RS aisle, there is one horizontal platform, whereas there is only one vertical platform serving the entire rack. Therefore, most of the time the horizontal platforms would be idle. If the horizontal platforms can pre-process jobs according to the incoming job sequence such that the VP waiting time for HPs is minimized, the handling time for each operation would be significantly reduced. This is the basic idea of the *preprocessing* scheme. A dispatching algorithm that enables both HPs and VP to detect their jobs and pre-respond accordingly before the jobs arrive is presented as follows.

```
PREPROCESSING :
  (1) set time window size $s$
  (2) scan the incoming jobs with the time window to collect jobs to
      be pre-processed.
  (3) for all jobs in a same batch, execute the following:
      (a) for retrieval job, the HP pre-fetches the requested container
          and waits at the handover stations whenever it is free after
          finishing the previous job; the container will then be trans-
          ferred to the VP, lowered down and roll into the I/O station
          whenever the VP is free after finishing its previous job.
      (b) for storage job, the HP is reserved at the transfer point and
          the VP at ground tier whenever they are available after fin-
          ishing their previous jobs.
  (4) advance time window and go to Step 2.
```

Job	Type & Destination	Job	Type & Destination	Job	Type & Destination
	(tier, bay)		(tier, bay)		(tier, bay)
1	R at (5, 8)	6	R at (4, 5)	11	R at (9, 13)
2	S at (6, 8)	7	R at (10, 3)	12	S at (8, 3)
3	R at (10, 9)	8	S at (8, 12)	13	R at (4, 13)
4	R at (3, 11)	9	R at (12, 3)	14	S at (12, 9)
5	S at (12, 3)	10	R at (4, 2)	15	S at (9, 4)

Table 4.8: Part of the generated job sequence

To examine how the scheduling scheme improves the throughput performance, we ran an example of 10,000 randomly generated jobs with and without pre-processing and collected statistics data on job handling time for analysis. The inter-arrival time of jobs followed Poisson distribution with different mean values varying from 30 seconds to 240 seconds. The first 15 jobs od the sequence are listed in Table 4.8 for example (S means storage; R means retrieval).

We executed the 10,000 jobs with and without preprocessing and collected statistics data on job handling time for analysis. The results are shown in Fig. 4.8, which indicates that the improvement on job handling time is significant when the schedule of jobs is not too tight. Note that the improvement in throughput might change with different job sequences. If the destinations of incoming requests are distributed more evenly, we can expect to see even higher improvement. Particularly, for a given job inter-arrival distribution, the best case for the proposed scheme is that individual jobs in the time window are located at different tiers, where the horizontal platforms could concurrently work to the maximum extent. In this case, if the mean value of the inter-arrival distribution is larger than the time it takes for the platforms to pre-process the jobs, the SP-AS/RS response time to each job would be the cycle time of the yard crane that transfers load between the I/O station and the prime movers.

Assuming an inter-arrival time of 120 seconds, consider a detected job sequence: $R(12, 14) \Rightarrow R(11, 14) \Rightarrow R(10, 14) \Rightarrow R(9, 14) \Rightarrow R(8, 14)$. The destination of the jobs are evenly

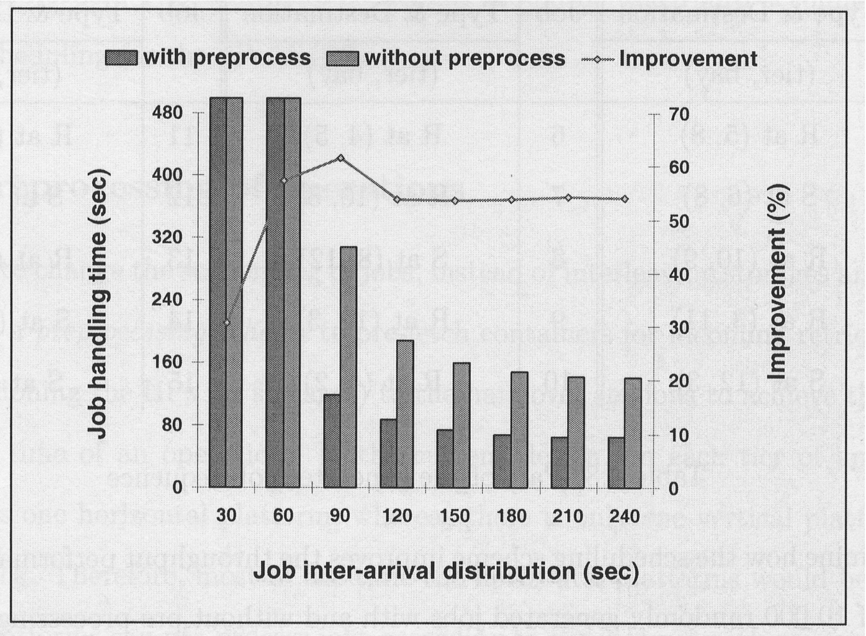

Figure 4.8: Improvement on job handling time

distributed and the HPs can pre-fetch the containers to the hand-over stations before jobs arrive. This is one of the best possible cases in which the preprocessing can achieve an improvement of 75.9%.

However, if the incoming jobs are concentrated on the same tier of individual SP-AS/RS racks, the HPs might be too busy to preprocess jobs and consequently the improvement will be reduced. The worst case would be such that all the detected jobs are retrieval and their destinations are all located in the highest tier. In this case, no improvement could be achieved if the mean value of inter-arrival of jobs is less that the time it takes for the HP to fetch the containers from the destination cells.

4.5.2 Basic idea of caching

To make the throughput improvement sequence-independent, we consider another scheme in which a buffer area near the I/O station is reserved in each SP-AS/RS rack. The platform will pre-store requested containers into the buffer area for detected retrieval jobs

Figure 4.9: A sketch of a storage tier with cache cells

and reserve empty cells in this area for detected storage jobs. If the buffer area is optimally designed, the SP-AS/RS response time will again be dominated by yard crane time only, regardless of the destinations of the incoming jobs. The algorithm is inspired by the cache structure in computer design.

We instigate an "Inner-tier caching" scheme, where a number of cells close to the hand-over station on each tier are reserved as cache cells. Incoming and outgoing containers are temporarily stored in these buffer cells before they are transferred to the destinations at proper time stamps, i.e., when the system is idle or when the retrieval vehicle arrives. The advantage of this scheme is that the caching operation on one tier does not interfere with that on another tier. Moreover, the vertical platform is not involved in the caching operation, which means a caching operation can be done even when an operation is in progress. A storage tier in this scheme is shown in Fig. 4.9.

The objective of this scheme is to minimize the response time of SP-AS/RS. Note that the average response time of cache cells would be much shorter than that of the memory cells. We define a storage hit as an empty cell found in the cache for the next storage operation, and a retrieval hit as the wanted container found in the cache. Then the "hit ratio" can be calculated as:

$$R_h = \frac{\#hits}{\#jobs}$$

Now the objective is to increase the hit ratio, which in turn can be achieved by fine-tuning the time window to detect the incoming jobs, by adequately scheduling the detected jobs, or/and by properly designing the cache size and the memory size.

The following is a dispatching algorithm that enables the horizontal platforms to detect their jobs and pre-respond accordingly well before the jobs arrive:

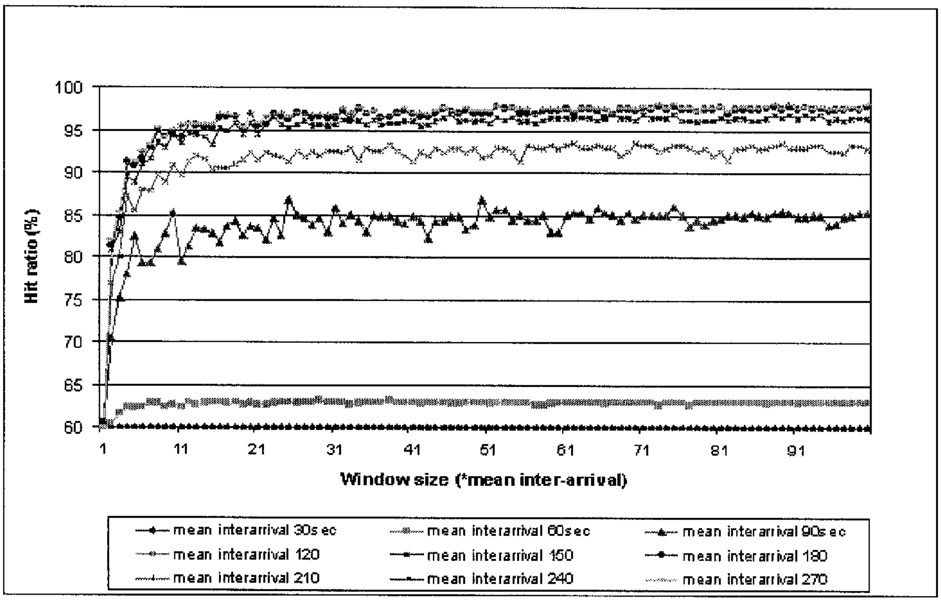

Figure 4.10: The improvement on hit ratio by window size

CACHING :

(1) scan the incoming jobs sequence with predefined time window.

(2) for the detected jobs, perform the preparation operations: empty cache cells for storages and shuffle boxes to cache cells for retrievals.

(3) advance time window and go to Step 1.

Note that in Step 2, the algorithm for scheduling the detected jobs plays an important role in deciding the performance of the scheme. A series of experiments have been conducted to test the improvement in the hit ratio, the results of which are summarized in Fig. 4.10.

As can be seen, without caching the hit ratio is around 60%, which predicts a poor response performance. Whereas when caching is performed, the hit ratio increases drastically to a level of 80% to 98%, depending upon the mean inter-arrival time of jobs and the size of the window. It is intuitive that when the inter-arrival time is longer, the HP would have more time to execute the caching operations, which in turn improves the performance. The confirms the effectiveness of the scheme. On the other hand, the hit ratio also increase with the size of the time window, which is again, consistent to intuitions. The more incoming jobs are detected, the more preparation operations can be done. Ideally the hit ratio should reach 100% if the inter-arrival time is long enough and the time window is large enough. However, this was not see in the experiments. To find the possible reasons, we further investigated the problem by classifying the causes "miss" into four categories as follows:

(1) Memory cell full: happens in a retrieval operation when the requested item is in a memory cell and requires swapping while the cache and the memory cells are all occupied. The swapping cannot be done in this case because no empty cell is available for temporarily storing the item. This could be resolved with optimal design of memory size and cache size.

(2) Limited time available: the inter-arrival time decides whether the HP would have enough time to perform the caching operations.

(3) Long execution time: there will be no time left for caching operations when the execution time of an operation is longer than or equal to the inter-arrival time.

(4) Fist job in the batch: caching can never be done for the first job in a batch.

To minimize the negative impacts of these problems, possible solutions could be: a well-designed tier configuration to eliminate misses due to cause 1, a well-designed time window to eliminate misses due to cause 4, and hopefully a better scheduling algorithm to reduce misses due to causes 2 and 3. Experiments have been conducted to quantify the misses related to the four causes and results are summarized in Fig. 4.11.

As shown, the misses due to time-related causes (2 and 3) can be largely reduced when the inter-arrival time increases. However, a "perfect" scheduling that gives least misses has yet to be found for the tight inter-arrivals. On the other hand, misses due to cause 1 could be reduced, if not eliminated, by designing an optimal rack layout. Fig 4.12 demonstrates the possibility. Apparently, cause 4 is unavoidable unless the time window is extremely large to detect all the incoming jobs which is unrealistic in practice.

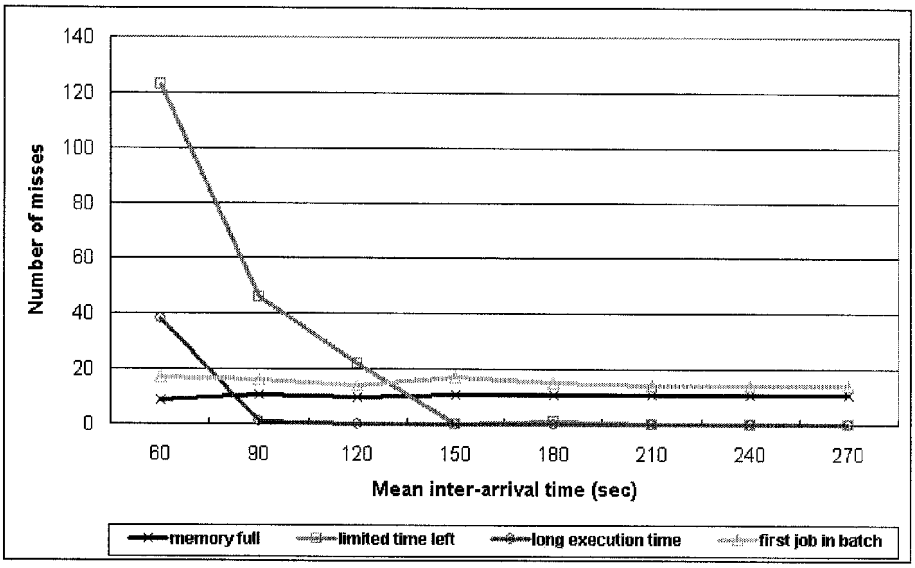

Figure 4.11: Classifying miss causes (window size fixed to 10)

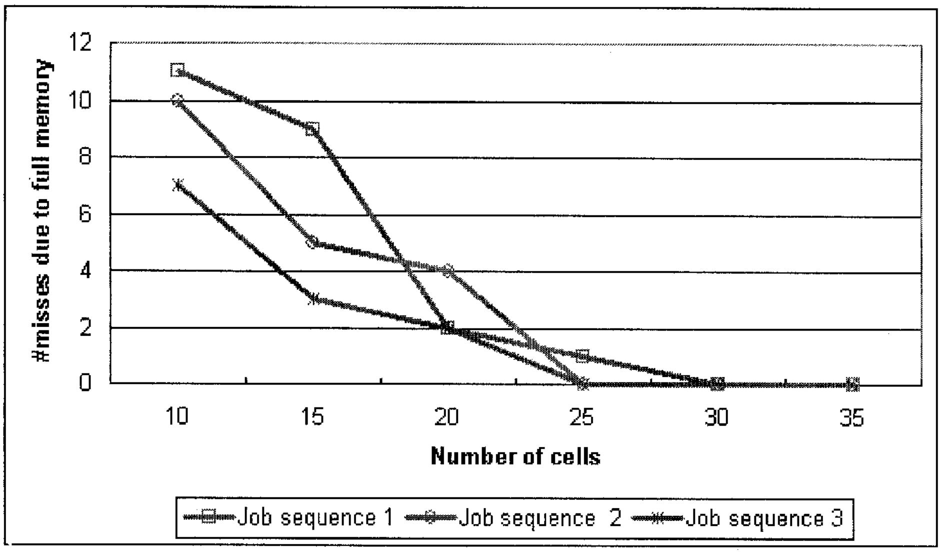

Figure 4.12: The impact of tier layout on memory-full misses (cache size fixed to 5)

Note that although the caching scheme is job sequence-independent, the time it takes for the HPs to perform caching and de-caching operations is considerably long, which requires a good estimate on the inter-arrival of jobs to each HP to ensure that the caching and de-caching operations would not interfere with the incoming jobs. This is, however, not easy to achieve in a container terminal with busy traffic. Moreover, the time spent on caching operations would consume additional energy, which contributes a big portion of operating cost. All these issues should be taken into account when deciding a scheduling scheme for SP-AS/RS in container operations.

4.6 Summary

In this chapter we have proposed a Split-platform AS/RS for container storage. The proposed system deploys horizontal and vertical platforms to transfer containers for safety and efficiency considerations. In order to evaluate the throughput feasibility of the system, we have developed discrete and continuous travel time models to preliminarily estimate the expected operation time. Two dwell point policies were also discussed, namely *residing* policy and *returning* policy, to determine the locations to posit the platforms when idling.

For the residing policy, we first gave the discrete expression of the expected operation time, which was validated by simulation results. The discrete models can accurately describe the platform behaviors and are computationally feasible. However, the expressions itself is rather complicated and to some extent, not practical for quick estimation of the operation time. In response to this drawback, we approximated the discrete rack as a continuous pick face to simplify the problem and set up simple continuous expression for the operation time with integration technique. The models were validated by simulation results. We found that the continuous models are applicable to estimate the expected operation times for the storage rack as long as the level of rack discreteness is carefully taken care of. It was observed that the global minimum of the expected operation time is obtained when the fraction of storages in the operation sequence is 0.5 and the shape factor is around 1.05.

For the returning policy, the discrete expression is straightforward and simple, due to the fact that both storage and retrieval operations will start from a known position. Similarly, the continuous expression is also simple and its accuracy is independent on the fraction of storages in the operation sequence. It appears that the rack has the minimum expected operation time when the shape factor is about 1.236.

Comparing the two policies, we found that for a balanced system, i.e., the inbound work-flow is equal to the outbound work-flow, the residing policy outperforms the returning policy. However, during certain time slots when the system works under an unbalanced situation, the latter policy can be considered to yield better throughput performance. Moreover, the continuous models tend to overestimate the operation times, it's therefore safe to estimate the throughput of the rack with these models, in which the overestimated part of operation times can be regarded as a safety factor.

We also proposed two algorithms for scheduling operations to further improve the productivity. The *preprocessing* scheme pre-fetches containers to I/O stations for retrievals and HPs to hand-over stations for storages, so that the response time to incoming operations can be significantly reduced. However preprocessing may not work when the jobs are concentrated on certain tiers in which case the HPs could not find available time slots to do the pre-fetching. Therefore the *caching* algorithm was proposed. Inspired by the cache and memory structure of computers, the algorithm pre-stores requested containers into a buffer area near hand-over stations for detected retrieval jobs and reserves empty cells in this area for detected storage jobs. As the operation time from buffer cells is much shorter than from other cells, the scheme manages to improve the throughput performance. Further examinations were taken to identify the causes of storage and retrieval miss in the cache and possible solutions were suggested to make the scheme more workable.

Moreover, the proposed SP-AS/RS has been tested in a simulation model that integrates each and every aspect of container port operations, e.g., berth allocation, quayside operations, and vehicle dispatching. The key performance indicators of container terminal operations were collected to evaluate the concept in an integrated view. The simulation runs yield a BOA rate (berth-on-arrival) of 91.9%, a QC rate of 43 lifts per hour, and a vessel rate of 109 lifts per hour. Meanwhile, the average vehicle waiting time at I/O stations is 90.6 seconds for storages and 134 seconds for retrievals, where the average yard crane cycle time is set to 80 seconds.

Chapter 5

Crane Deployment for AS/RS Blocks

We have assumed that there are enough yard cranes to serve the requests from the I/O stations of the AS/RS in the previous chapters so as to simplify the analyses. It is, however, more realistic to acknowledge that in a container terminal that the storage blocks share a certain number of yard cranes, which discharge and load containers from/to the vehicles. The efficient use of the limited number of yard cranes becomes, in this case, another key to a satisfactory performance in a terminal. This involves the study on yard crane deployment and scheduling. In this chapter, we investigate this problem by modelling it as a $k-$server problem that is extensively explored in the online algorithm context, and propose practical online algorithms. Note that the algorithms are also applicable for deploying yard cranes in a conventional stacking yard.

5.1 Statement of the problem

To implement AS/RS for container storage, an effective and practical method is needed for scheduling the cranes to quickly respond to the requests mounted at the I/O stations, so as to take advantage of the increased productivity offered by AS/RS.

Consider N storage blocks, each of which consists of a certain number of AS/RS aisles, and K yard cranes (YCs) are available in the yard to serve the N blocks. The crane deployment problem is to assign tasks mounted from the I/O stations to individual cranes such that certain objectives are achieved. This includes inner-block deployment and inter-block deployment. Assume that all the K cranes are dedicated to serve a particular block

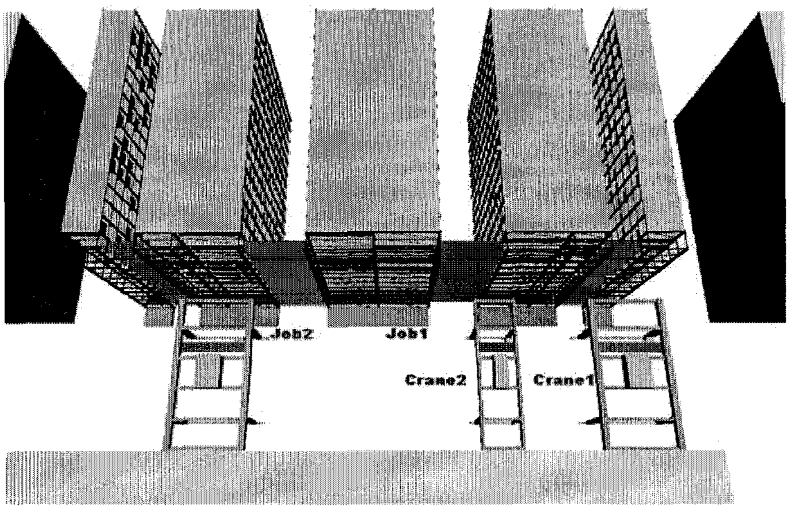

Figure 5.1: Bird's view of an AS/RS storage block

as shown in Figure 5.1. The K cranes share a common bi-directional travelling lane on which they can make liner movement to serve any of the N I/O stations with outstanding jobs. Overtaking is prohibited since the lane is wide enough to accommodate one crane only. In other words, a crane on left can never serves a job on the right of another crane, e.g., crane 1 in Figure 5.1 is unable to serve job 1 unless crane 2 moves to job 2. The inner-block deployment is to decide, when requests mounted on an I/O, which particular crane to serve the requests.

A generalization of the problem is to allow the K cranes to serve other blocks in a layout as illustrated in Figure 5.2. Crane $C1$ in block $B1$ can travel to block $B2$ along a straight line or to block $B4$ by making 90 degree turns. This typically involves the inter-block deployment of cranes. Note that cranes are big and heavy, their gantries, especially inter-block movements, take a great amount of time and road space. Frequent crane gantries will obstruct the road traffic and result in loss in productivity. It is always desirable, in fact the objective of deployment, to keep the crane movements as few as possible.

Solutions to the deployment problem are expected to efficiently work in the context of container terminal operations, typical additional requirements from which are elaborated as follows.

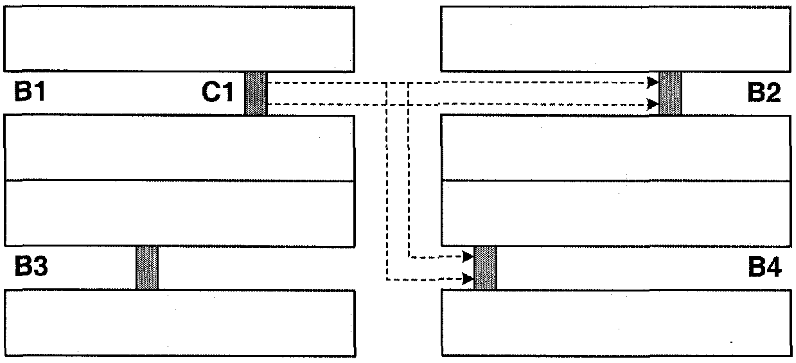

Figure 5.2: Crane gantries among blocks

5.1.1 Planning period

Container terminals work three 8-hour shifts round the clock every day. The resource deployment, e.g., the crane scheduling, is accordingly performed on a rolling planning horizon basis. There are several considerations for deciding the length of planning period. First of all, a schedule should be available before the beginning of each shift. Therefore a planning period should not span across a shift. Secondly, activities during the planning period should be predictable with an acceptable accuracy, taking into account the dynamic nature of container arrivals and departures. This imposes that the planning period should be long enough to improve the estimate accuracy. However, it is not reasonable to stretch the planning period for too long, as a crane that finishes its jobs early may be idling for a long time. In summary, to find a good planning horizon, the following conditions must be taken into consideration:

- Less than or equal to the length of a shift.
- Workload can be estimated with an acceptable accuracy period as long as possible.
- Workload should be distributed as uniformly as possible period as short as possible

5.1.2 Input data

The information necessary for planning includes two types, namely static and dynamic data. Static data are the constant parameters such as the number of blocks and the number of cranes. Dynamic data are normally related to yard status that are updated in real time, e.g., the current positions of cranes.

The arrivals and departures of containers trigger the changes in the dynamic data. The volume and expected time of arrival (ETA) of containers to the storage blocks from vessels can be estimated based on the quayside and yard-side planning such as berthing schedule and yard allocation. On the gate side, however, the ETA of jobs are unpredictable as in most ports, customers are allowed to drop off and pick up containers without making reservations. They can only be forecast based on the historical data, which requires that the planning period must not be too short to maintain the accuracy level of the estimate on workload for storage blocks. To summarize, the input data available for making crane scheduling decision include:

- W_b: workload forecast for block b, measured in number of storage/retrieval jobs. This includes:
    - WS_b: ship operations.
    - WG_b: gate operations.
    - WP_b: workload leftover from the previous period.

    $W_b = WS_b + WG_b + WP_b$
- RI_b: number of YCs in block b at the beginning of a period, as an initial condition.
- Given conditions:
    - P: number of jobs a YC expected to finish within the planning period.
    - T_{ij}: YC travel time between blocks i and j.

5.1.3 Constraints

There are also constraints to the schedule, which are imposed either by resource availability or equipment operation requirement. For instance, due to the limited size of a storage block and the clearance requirement on adjacent cranes, only a certain number of cranes are allowed to work simultaneously in a block. Two main constraints considered in the study are:

- maximum number of YCs allowed in a block, R_{max}.
- maximum number of YCs to be deployed in a shift, R_{total}.

5.1.4 Objectives

The main objective of the scheduling would be improving crane productivity to minimize the workload leftover to the next shift that decreases the servicing level of a terminal. However, note that the processing times of the jobs would sum up to identical for different schedules, therefore it is the minimization in crane gantry movement that distinguishes an optimal schedule. Nevertheless, the objective of a scheduler is to:

- minimize total workload leftover: $\sum_{b=1}^{B} O_b$, or
- minimize the sum of distances travelled by the cranes: $\sum_{k=1}^{K} G_k$

5.2 Related works

Scheduling of multiple yard cranes has not been systematically studied, despite its importance to the day-to-day operations in container terminal. Only several recent research addressed this issue, most of which formulating it as an integer program or a mixed integer program and solve with various techniques. In particular, to find an optimal deployment scheme for K YCs and B blocks, the minimization of the amount of workload leftover in one period is normally identified as objective, with the amount of YC inter-block movements as decision variable. Without loss of generality, the following definitions are introduced:

- A block: a block is an *A*ctive block if its forecast workload $W_b \neq 0$.
- U block: a block is an *U*nderflow block if it has underflow workload satisfying $U_b = RI_b \cdot P - W_b > 0$.
- O block: a block is an *O*verflow block if it has overflow workload satisfying $O_b = W_b - RI_b \cdot P > 0$.
- E YC: a YC is an *E*ligible YC if it can be deployed in the current shift.
- M YC: a YC is a *M*ovable YC if it can finish its jobs in its current block early and thus can be moved to other blocks.

Zhang et al. [ZWLL02] constructed an MIP model to find the optimal movements of cranes, including designation of cranes for inter-block movements and the corresponding timings. The formulation estimated W_b at the beginning of each planning period and assumed that inter-block movement of each crane is allowed only once and must be completed during one period. Lagrangean relaxation was applied to solve the problem, with modifications to improve the solution quality by introducing additional constraints. Similar idea was proposed by Cheung et al. [CLL02], who introduced a successive piecewise-linear approximation method to solve the problem iteratively, in each step taking it as a linear network flow problem. The method removed the restriction that crane movement must be done within a period. Linn et al. [LLW$^+$03] suggested a simpler formulation for the scheduling problem, in which E cranes and O blocks are first identified before assigning M cranes to O blocks in the deployment step. The assignment was confronted with constraints such as one crane moving only once, and solved by constraint programming technique, in particular implemented in CPLEX. W.C. Ng. [Ng04] considered the crane scheduling problem within a single linear zone where a crane can move from one block to another without making turns (e.g., from $B1$ to $B2$ as in Fig. 5.2). A decomposition approach was proposed to partition the K crane scheduling problem into K independent single crane scheduling problems. Assuming that the arrivals of jobs were known beforehand, a dynamic programming method was used to determine an effective partition, which was followed by a job reassignment procedure to make up for possible missing-out of good schedules due to the partition.

Although not stated, the following assumptions are necessary for the models in the above studies:

- All YCs can work on the jobs without waiting. In other words, all W_b workloads are available from the beginning of each period.
- Vehicles can wait for arbitrarily long period of time until a YC comes to serve.
- No ordering requirement is considered for all jobs. In other words, a YC can serve the jobs from different blocks in any order.
- Cranes in a block can always work simultaneously without interference to each other (except in [Ng04]).

These assumptions are, however, unrealistic and potentially prevent the algorithms being implemented in practice, due to the following reasons:

- Vehicle arrivals spread over the planning period: An E YC at a U block is not necessarily an M YC, or it may have to make frequent unproductive inter-block gantries.
- The queuing space for vehicles is limited: An M YC may have to move to a heavily congested block before it finishes its current jobs and consequently alternate the 'optimal' schedule.
- During loading phase the containers must be retrieved in a predefined order: A YC cannot just let vehicles in other blocks wait until it finishes its current jobs (defined by the 'optimal' schedule), if the jobs are in fact with lower priority according to the loading list.
- A crane may be forced to stay idle simply because of its interference with another crane. The time needed to finish a particular amount of workload in a block is not proportional to the number of cranes.

Therefore, taking into account the dynamic nature of job arrivals, an online algorithm is preferred where the knowledge of the incoming job requests is not required for planning.

5.3 Event-triggered crane scheduling

We shall first discuss how to address the above mentioned issues when designing an scheduler for practical application. First of all, the arrivals of jobs are unknown to the scheduler, which can only make use of past requests, if remaining outstanding at current time. On account of this, the scheduler consists of an online algorithm that takes real time input and produces real time commands. Secondly, considering that the queueing space for vehicles is limited, a threshold is introduced to the scheduler to maintain the queue lengths in individual blocks. Thirdly, on the precedence requirement, an event is generated for a new incoming job with a higher priority to make it jump the queue. And lastly, safety clearance and non-crossing constraints are introduced into the scheduler when assigning YCs to jobs, so as to take care of the crane interference. With all these elements included, the scheduler is able to tackle all the issues by events, which is therefore named as event-triggered crane scheduling.

Note that, however, the scheduler will consider inter-block deployment of crane only when a queue is built up in a block without cranes, i.e., a block with working cranes will not generate any inter-block deployment events even if it has a long queue. This is based on the observation that sending a crane to a block with working cranes may not be able to help ease the congestion, due to the unpredictable job arrivals and the possible interference between YCs. For instance, sending a crane to block $B1$ in Figure 5.2 is pointless because the crane is not able to share loads with the existing crane $C1$ when all jobs are queuing for where $C1$ is. Moreover, even if in some cases it works, the operation time saved could be offset by the time it takes to make the inter-block movement.

The procedure of scheduling is described in the following subsections, with the following additional definitions:

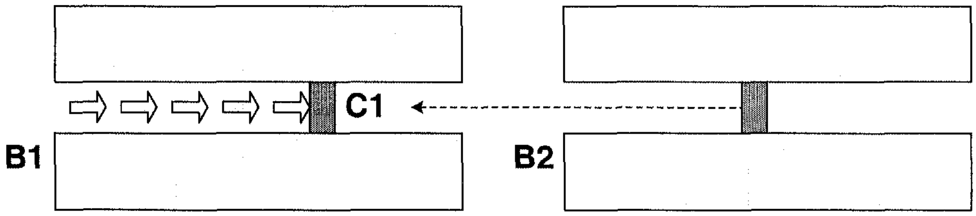

Figure 5.3: Illustration on non-inter-block movement event

- B block: a block is a Busy block if it has forecast ship operations, i.e., $WS_b \neq 0$.
- F blocks: any two adjacent blocks are Friend blocks if they align longitudinally, e.g. $B1$ is an F block for $B2$ and $B2$ is an F block for $B1$ in Figure 5.2.

5.3.1 Procedure

(1) Establish the number of B blocks, N_B, and the number of A blocks, N_A.

(2) Establish the number of YCs needed for each block, $R_b = ceiling\{W_b/P\} \leqslant R_{max}$.

(3) Calculate the number of YCs needed for the shift, $R_{need} = \sum_{b=1}^{B} R_b$.

(4) Compare R_{need} and R_{total}:

A. $R_{total} \geqslant R_{need} \Rightarrow$ #YCs needed can be satisfied, no inter-block YC movement is needed within a period:

A.1 Identify E YCs and M YCs with function **markYC**().

A.2 Find M YCs for O blocks with function **findYC**().

A.3 Park extra YCs, if any, with function **parkYC**().

A.4 In the blocks where $R_b > 1$, assign incoming jobs to YCs with algorithm THRESHOLD-RDC .

B. $R_{total} < R_{need} \Rightarrow$ only R_{total} YCs can be used, which must be shared during the period:

B.1 Park extra YCs, if any, with function **parkYC**().

B.2 Compare N_A and R_{total}:

(a) $R_{total} \geqslant N_A$, do function **allocYC**$\|_{\textbf{SORT}}()$.

(b) $R_{total} < N_A$, compare N_B and R_{total}:

i. $R_{total} \geqslant N_B$, do function **allocYC**$\|_{\textbf{SHARE}}()$.

ii. $R_{total} < N_B$, do function **allocYC**$\|_{\textbf{ONLINE}}()$.

5.3.2 Functions in the procedure

markYC(): This function identifies E YCs and M YCs according to the initial conditions in the blocks.

(1) Set $b = 1$.

(2) If $RI_b > R_b$ then set the first R_b YCs in block b as E YCs and the rest YCs as M YCs; otherwise set all the RI_b YCs as E YCs.

(3) Increment b by 1 to mark YCs in next block until $b = N$ (N is the number of blocks).

findYC(): This function finds YCs for the O blocks from U blocks.

(1) Identify friend blocks for each O block.

(2) Search for M YCs from blocks in the following order: friend block, section blocks, YC park, and the remaining blocks.

parkYC(): This function parks the additional YCs in the blocks in appropriate positions so as not to interfere with E YCs.

(1) Park YCs in their current positions if the current block is not an A block.

(2) Check the working area in the next shift for each A block with extra YCs.

(3) Park as many extra YCs in the current block as possible at either edge of the block, without interfering operations.

(4) Send the rest extra YCs to YC park.

allocYC$||_{\text{SORT}}$(): This function deploys YCs according to the workloads in the blocks. The workloads related to ship operations are considered with higher priority and thus more, if any, YCs are assigned to a block with bigger WS_b. Note that during the shift, the YCs will not be re-deployed for inter-block movements.

(1) Sort the A blocks by WS_b, tie-break with W_b.

(2) Assign each A block with 1 YC.

(3) Assign the remaining YCs one by one to the sorted A blocks.

allocYC$||_{\text{SHARE}}$(): This function deploys YCs with higher priority to B blocks. The rest A blocks share the remaining E YCs.

(1) Assign each B block with 1 YC. These YCs will not be re-deployed during the shift.

(2) The rest $A - B$ blocks are allocated with the remaining $(R_{total} - N_B)$ YCs.

(3) Within the planning period deploy the $(R_{total} - N_B)$ YCs with THRESHOLD-HARMONIC.

allocYC$||_{\text{ONLINE}}$(): This function shares all E YCs among the A blocks.

(1) Sort the A blocks by WS_b, tie-break with W_b.

(2) Assign one by one the YCs to the sorted A blocks.

(3) Within the planning period always deploy YCs with THRESHOLD-HARMONIC.

In the next sections, we will elaborate in details the two algorithms used in the deployment procedure: THRESHOLD-RDC and THRESHOLD-HARMONIC.

5.4 Scheduling algorithms

We start the section with an introduction on a related problem that has been extensively studied in online algorithm context, the $k-$server problem, which has been the inspiration and the source of the two algorithms proposed.

5.4.1 The $k-$server Problem

The **k-server problem** is a natural generalization of the paging problem that also encapsulates a wide range of scheduling problems such as weighted paging and multi-threaded disk scheduling. It was first introduced by Manasse, McGeoch, and Sleator [MMS90], and can be formulated as follows [BEY98]:

Let (M, d) be a metric space where M is a set of points and $d : M \times M \longrightarrow \Re$ is a distance function on these points so that for any three points $x, y, z \in M$:

i. $d(x, y) \geqslant 0$ iff. $x = y$

ii. $d(x, y) = d(y, x)$

iii. $d(x, z) \leqslant d(x, y) + d(y, x)$

Let k be an integer and $|M| > k$. The **k-server problem** consists of managing k mobile servers over M to serve a sequence $\sigma = r_1, r_2, \ldots, r_n$ of requests where a request r_i is a point in the space. An algorithm must, at each step, ensure that the requested point is covered by a server. The cost of the algorithm is normally, but not necessarily, defined by the sum of distance travelled by all the servers.

An online algorithm is given the current and past requests from σ, based on which it decides how to move the servers. In other words, it will only know r_{i+1} when r_i has been served.

Summary of known results

1. *The optimal off-line algorithm and the $k-$server conjecture*

   If the full knowledge of σ is known when an algorithm schedules the server movement, the schedule is called an off-line schedule. The optimal schedule can be computed by using dynamic programming, or by formulating it as a min/max flow problem [BEY98].

For the online version, Manasse, McGeoch, and Sleator formulated the famous $k-$server conjecture: *For any k and for any metric space, there is a deterministic k−competitive algorithm* [MMS90]. The conjecture has been almost solved after considerable effort when Koutsoupias and Papadimitriou proved that the Work Function Algorithm (WFA) is $(2k-1)-$competitive in any metric space [KP95a].

2. *Deterministic algorithms for specific k*

$k = 2:$ Both RES algorithm [CL91a] and WFA [KP95a] are 2-competitive. There are also a great amount of work on developing "fast" and competitive algorithms for 2-servers, e.g., the SLACK-COVERAGE algorithm [Bar94].

$k = |M| - 1:$ The BALANCE algorithm due to [MMS90] is $k-$competitive.

3. *Deterministic algorithms for specific metric spaces*

For the Euclidean line and tree spaces, there is a $k-$competitive algorithm called DOUBLE-COVERAGE or DC for short [CKPV91]:

Algorithm DC : If the requests falls outside the convex hull of the servers, serve it with the nearest server. Otherwise, the request is in between two adjacent servers. In this case, move both servers toward the request at equal speeds until (at least) one server reaches it (if two servers occupy the same point, then choose one arbitrarily).

The algorithm can be naturally generalized to trees as follows [CL91b]:

Algorithm DC-TREE : At each time, all the servers neighboring the request are moving in a constant speed toward the request until one server reaches it.

4. *Randomized algorithms*

It was realized that randomization could possibly offer the online algorithm significantly more power, as the decisions on server movement by the online algorithm are no longer certain (to the adversary) [BEY98]. The power of randomization was investigated against oblivious adversaries and adaptive adversaries [BDBK+94]. For $k-$server problem, the HARMONIC algorithm was found to be constant-competitive with respect to the number of requests by Grove [Gro91].

> **Algorithm HARMONIC :** Serve an uncovered request r with a randomly chosen server where each server is chosen with probability inversely proportional to its distance to the request node.

Recall that the deterministic DC algorithm is competitive for line. It can also be made memoryless and randomized as follows [CDRS93]:

> **Algorithm RDC :** If the request r_i falls outside the convex hull of the servers, serve it with the nearest server. Otherwise, the request is in between two adjacent servers, e.g., s_1 and s_2. Serve r_i using server s_j ($j = 1, 2$) with probability $\frac{1/d_{s_j,r_i}}{1/d_{s_1,r_i}+1/d_{s_2,r_i}}$, where d_{s_j,r_i} is the distance between request r_i and server $s_j, j = 1, 2$.

5.4.2 Inner-block deployment: THRESHOLD-RDC

Consider each YC as a server on a line and each I/O station as a point of the line and factor the operation time on a request into crane gantry time. We can regard the problem as $k-$server on a line and make use of the RDC algorithm to assign jobs to multiple YCs in a block, by setting a predefined threshold, reaching which would trigger an event to deploy a YC to serve the requests. In particular, the algorithm works as follows:

THRESHOLD-RDC :

(1) for each request mounted at an I/O station, convert it to workload with ship request having a weight bigger than 1.

(2) position YCs at their current locations to serve the requests until one of the following conditions met:

  (a) the workloads at an I/O, e.g. I/O m, reaches a predefined threshold l, i.e., $W_m \geqslant l$.

  (b) a request with precedence requirement arrives at an I/O without server.

(3) find one YC to serve I/O m with algorithm RDC.

(4) go to step 2 until all requests served.

Note that when setting the threshold l, the physical constraint arose from the available queuing space between I/O stations must be taken into account.

5.4.3 Inter-block deployment: THRESHOLD-HARMONIC

In the inter-block deployment case, we again regard each block as one point on the metric space, and each YC as a server in the $k-$server problem. The algorithm sends the R_{total} YCs to serve the first R_{total} blocks, sorted by the amount of workloads, until a threshold l is reached by one of the blocks without servers (YCs). The algorithm then chooses one of the YCs with HARMONIC algorithm to serve the block. In particular, the algorithm works as follows:

```
THRESHOLD-HARMONIC :
  (1) sort the blocks by $WS_b$, tie-break with $W_b$.
  (2) position all available YCs one by one to the sorted blocks, i.e.,
      the first $R_{total}$ blocks.
  (3) for each request generated from a block, convert it to workload
      with ship request having a weight bigger than 1.
  (4) serve the requests from the $R_{total}$ blocks until one of the following
      conditions met:
      (a) the workloads of a block, e.g. block $m$, reaches a predefined
          threshold $l$, i.e., $W_m \geqslant l$.
      (b) a request with precedence requirement arrives at an I/O
          without server.
  (5) find one YC to serve block $m$ with algorithm HARMONIC.
  (6) go to step 4 until all requests served.
```

Similarly, the setting of threshold l should take into account the queuing space available for the blocks.

5.4.4 Discussion on the dispatching algorithms

The efficiency of the proposed THRESHOLD-RDC and THRESHOLD-HARMONIC can be tested against a greedy algorithm, which always sends the nearest crane to serve the mounted jobs. The major flaw in the greedy algorithm is that if a large number of jobs concentrate on one or two sides of the convex hull of the cranes, e.g., on the left hand side of all the cranes,

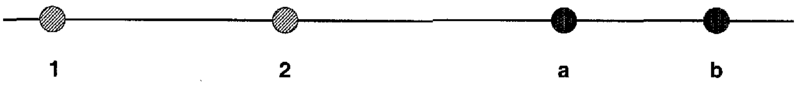

Figure 5.4: A 4-node graphic showcasing the uncompetitiveness of Greedy

then greedy algorithm always send the left-most crane to serve the jobs while maintaining other cranes idle. Consider, for instance, a 2-crane problem when the metric space is the 4-node graph shown in Figure 5.4 and the sequence of jobs is *ababab*.... It is obvious that the greedy algorithm will attempt to serve all the jobs with crane 2 incurring an unbounded cost. In such cases, a low productivity can be anticipated.

Interestingly, in container storage blocks such consolidation of operations would happen very often. This is because of the characteristics of the import blocks and the export blocks in a container yard. The import blocks are normally filled up with a particular pattern, e.g., from left to right, that is decided by the yard allocation scheme. Such a pattern would introduce consolidated arrivals of vehicles, be they coming for storage or for retrieval. On the other hand, the export blocks are segregated in clusters, each of which is designated for a particular vessel. Although the arrivals of vehicles for storing containers in the clusters can be spread (within a period of time), the retrievals of containers would concentrate during the loading phase of the vessel. Therefore the above example would very likely happen often in a container terminal, which implies the advantages of the proposed algorithms over the greedy algorithm.

To showcase the improvement in a quantitative manner, we conducted a number of simulation runs with different inter-arrivals of jobs, and collected as criterion the percentages of jobs finished by THRESHOLD-RDC and by greedy algorithm. The algorithm for the simulations is summarized in next page in "Simulation Algorithm". We considered the 2-crane serving one storage block case, assuming there are 24 I/O stations within the block for the cranes to perform jobs (refer to 5.1). 1,000 jobs were executed in each experiment by the two cranes. The job locations were randomly selected from the 24 I/O stations and considered independent of the queuing status (that is, a job can be always be mounted at any I/O station, regardless of the number of jobs queuing at the I/O). 8 replications of simulation with different random seeds were executed for each layout, which were able to achieve a relative error of 0.01 at 95% confidence level.

```
SIMULATION ALGORITHM :

(1) read as input the mean inter-arrival time of jobs.
(2) generate a sequence of job with an attribute indicating the location, using Poisson distribution.
(3) schedule the cranes to execute the jobs by:
    (a) Greedy.
    (b) THRESHOLD-RDC.
(4) terminate the simulation at a pre-defined time
(5) collect the percentage of jobs done
```

The results are summarized in Figure 5.5. As can be seen, when the mean inter-arrival is very short, both greedy and proposed algorithm will have problem in handling the jobs quickly due to the limitation of the crane handling speed. Therefore little improvement of the proposed algorithm was observed. However when the inter-arrival increases and becomes long enough for the THRESHOLD-RDC algorithm to schedule the jobs, the improvement becomes noticeable and peaks at around 1.5 minutes, where the back-and-forth movement of cranes in greedy algorithm largely decreases the productivity and causes tremendous delays in handling jobs. When the inter-arrival keeps increasing and becomes long enough even for the greedy algorithm to schedule the jobs, the improvement becomes less significant and finally diminishes.

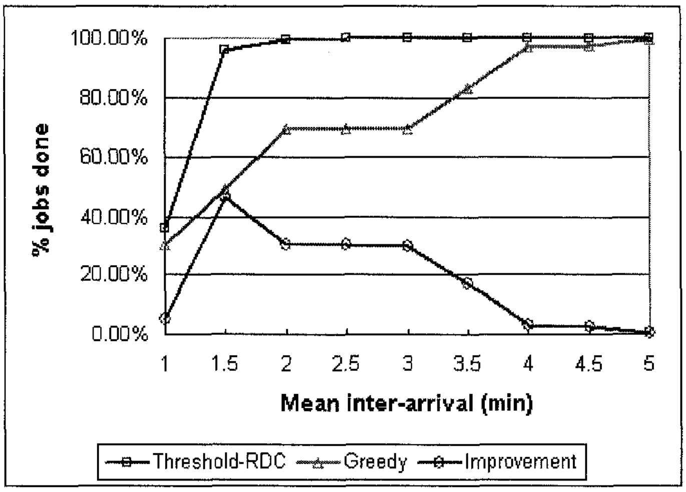

Figure 5.5: Advantage of THRESHOLD-RDC algorithm over Greedy

Recall that THRESHOLD-HARMONIC shares similar advantages with THRESHOLD-RDC over greedy algorithm. Both randomly choose a crane with probability related to the distance between the mounted job and the crane position, Harmonic using a probability distribution that is inversely proportional to the distance (Section 5.4.3), while RDC taking into account whether the position of the job falls within the convex hull of the cranes (Section 5.4.2). The randomness in both algorithms avoids the flaw that the greedy algorithm suffers (Figure 5.4). If we consider each block when doing inter-block scheduling by THRESHOLD-HARMONIC as an I/O station in THRESHOLD-RDC , we can expect similar improvement in THRESHOLD-HARMONIC .

5.5 Summary

In this chapter, we have discussed the importance of efficient scheduling algorithms for yard cranes and formulated the problem as the well-know $k-$server problem. Two algorithms, THRESHOLD-RDC and THRESHOLD-HARMONIC , were proposed for scheduling inner-block movements and inter-block movements of yard cranes, respectively. Their advantage over intuitive greedy algorithm was discussed. Note that the algorithms are also applicable for deploying yard cranes in a conventional stacking yard, because we have not introduced any particular assumptions specific for the AS/RS blocks.

Chapter 6

Concluding Remarks and Directions for Future Studies

In this thesis, we have addressed the necessity of introducing more efficient storage structures and better operational planning approaches for a container terminal, in response to the ever-increasing throughput demand and regional competition. Inspired by its abilities of fully automating material handling and the potentials for delivering high performance, we suggested AS/RS for container storage and proposed a platform-based scheme to take into account the additional constraints arising from the container size and weight. The operation time models were developed for the proposed system to preliminarily evaluate the performance feasibility. The dispatching of cranes that interface with container carriers is also studied in order to achieve a compatible crane service rate. The results indicate that application of AS/RS in container storage is a promising way of improving yard performance. In this chapter we will summarize the findings and indicate possible directions for future studies.

6.1 Concluding remarks

The purpose of this study is to investigate efficient storage structures for container terminals, which is critical to improve their performance and competitiveness. Aware of the advantages of AS/RS and its successful applications in other industries, we have proposed a concept of storing containers in split-platform AS/RSs (or SP-AS/RS in short). The tasks involve the optimal design of the structures and their corresponding planning and operational schemes.

We have proposed one design for the SP-AS/RS, for which we have developed discrete and continuous operation time models to estimate the performance of the system. Two dwell point policies for locating the idle platforms were discussed, namely *residing* policy and *returning* policy, which go into parts of the control policies. Moreover, we also proposed two algorithms for scheduling operations to further improve the productivity. The *preprocessing* scheme pre-fetches containers to I/O stations for retrievals and platforms to hand-over stations for storages to reduce the response time to incoming operations, while the *caching* algorithm, inspired by the cache and memory structure of computers, pre-stores requested containers into a buffer area near hand-over stations for detected retrieval jobs and reserves empty cells in this area for detected storage jobs to improve the throughput performance. Based on the results, we come to the following conclusions:

(1) For both of the dwell point policies, discrete models can accurately estimate the expected operation times of storage racks and are computationally feasible. However, the expressions themselves are complicated and and to some extent, not practical for quick estimation of the operation time.

(2) On the other hand, the continuous models are simple and expressed as a close-form function of the shape factor b and the fraction of storage operation in the job sequence α. Their accuracy is affected by the discreteness of the rack and generally slightly lower that that of the discrete models. Fortunately, in practice the size of the cells would be limited, e.g., $4.5m \times 4.5m$ for container (considerably large items) storage in our case, while the area of the rack could be more flexibly decided. The size of the rack in general is large enough for the continuous models to return good estimates. Therefore, if we pay attention to the rack discreteness when applying the continuous models, their accuracies can be assured at an acceptable level.

(3) The continuous models tend to overestimate the operation times, it is therefore safe

to estimate the throughput of the rack with these models, in which the overestimated part of operation times can be regarded as a safety factor.

(4) Under the *residing* policy, the minimum expected operation time is obtained when α is 0.5 and b is around 1.05. Under the *returning* policy, the rack has the minimum expected operation time when the shape factor is about 1.236, regardless of the value of α.

(5) For a balanced system, i.e., the inbound work-flow is equal to the outbound work-flow, the residing policy outperforms the returning policy, which can therefore be regarded as the preferable policy for container storage. However, during certain time slots, e.g., the vessel unloading phase, the system works under a unbalanced situation. In this case, the latter policy could be considered to yield better throughput performance.

(6) The *preprocessing* scheme is efficient when the sequence of jobs spreads over different tiers so that spare time of horizontal platforms can be used for the preprocessing operations. The improvement in the response time to a job mounted at the I/O station could be as much as more than 60%.

(7) The *caching* scheme imposes more workloads on the platforms but its improvement in productivity is independent of the job sequence. Ideally the response time could be as short as the cycle time of the interfacing yard crane, depending on the optimality of the design of the cache.

The testing results suggest that the application of AS/RS in container storage can offer dramatic improvement in container handling (a cycle time of less than 2 minutes compared to the current 4 minutes). When placed in a simulation model that integrates each and every aspect of container port operations, e.g., berth allocation, quayside operations, and vehicle dispatching, the key performance indicators of container terminal operations were found to be promising. In particular, it gave a BOA rate (berth-on-arrival) of 91.9% and a QC rate of 43 lifts per hour [Tea03]. This is an noticeable improvement over the performance of traditional terminals where the measures are 80% and 30, respectively [Tea03].

On the other hand, it was noted that when the drastic increase in the productivity of storage blocks has imposed additional requirements on the scheduling of yard cranes that interface between the I/O stations and the container carriers. We proposed two algorithms to efficiently schedule the cranes to quickly respond to the requests mounted at the I/O stations. The algorithms were tested against the intuitive greedy algorithm by simulation runs. The results show that they are efficient and more importantly, practical, for improving crane productivity.

6.2 Directions for future studies

As this thesis conducts only a preliminary study of the innovative approach, a lot of reinforcement on the system can be carried out in the following aspects:

(1) Other than randomized storage, more storage allocation policies should be examined to seek further performance improvement, e.g., class-based storage that is reported to be more efficient in conventional AS/RS.

(2) An optimal cache design for the *caching scheme* also deserves careful investigation, as it could lead to the ideal case where the response time of the AS/RS blocks is minimized.

(3) On allocating containers to storage racks, a compatible policy should be found, on one hand, to balance the workloads among the racks, and on the other, to minimize the total travel distance from the berth to the destination racks. The two objectives, in most cases, conflict with each other and therefore a tradeoff is desired.

(4) The online algorithms for the scheduling of yard cranes require further studies as well. For instance, a study on their time complexities and their competitiveness would be of most value for exploring possibilities of improvement.

Appendix A

Abbreviations

AGV	Automated Guided Vehicle
AS/RS	Automated Storage and Retrieval Systems
BOA	Berth-On-Arrival
CL	Closest Open Location
CO	Current Operation
DC	Double Coverage
ETA	Expected Time of Arrival
FCFS	First-Come-First-Served
HP	Horizontal Platform
I/O	Input/Output
LO	Last Operation
LWR	Longest Waiting Retrieval
NN	Nearest Neighbor
NR	Nearest Retrieval
NSR	Nearest Storage/Retrieval
OHBC	Overhead Bridge Crane
PO	Preceding Operation
QC	Quay Crane

RDC	Randomized Double Coverage
RMG	Rail Mounted Gantry Crane
RN	Random Open Location
RTG	Rubber Tyred Gantry Crane
SL	Shortest Leg
SP-AS/RS	Split-Platform Automated Storage and Retrieval Systems
S/R	Storage/Retrieval
TEU	Twenty-foot Equivalent Unit
VP	Vertical Platform
YC	Yard Crane

Appendix B

Proofs of Claims

Claim 4.3 *Storage time for case b2, where* $(x,y) \neq (x',y') \bigwedge (y \neq y')$ *is given by:*

$$t_s^{b2} = \frac{(B+1)(2T-3)}{4B(T-1)}y + x + \frac{(B+1)(T^2-5T+6)}{4B(T-1)}$$
$$+ \frac{1}{4B(T-1)} \sum_{y' \neq y} \sum_{x'' \neq x} (max(y+y'-2, x'') + max(y-1, x''))$$

Proof: In this case, the preceding operation is to a different tier. We first consider the case that the preceding operation was a storage, assuming that the last operation on tier y was applied to cell (x'', y). If $x'' = x$, then it must have been a retrieval with a probability of $1/B$. According to Property 4.1 and 4.2, the storage time for this subcase can be written as:

$$t_1 = (y'-1) + (y-1) + x$$

If $x'' \neq x$ (with a probability of $(B-1)/B$), the last operation on tier y could be a storage or a retrieval with equal probability, i.e., 0.5. The storage time if it was a retrieval has been given above. Similar to case b1, the storage time for the latter case can be calculated as:

$$t_2 = \frac{1}{B-1} \sum_{(1 \leqslant x' \leqslant B) \bigwedge (x' \neq x)} max((y'-1)+(y-1), x'') + x$$

Combining the above equations and weighting them with their probabilities, we have the storage time for the case where the preceding operation is a storage as follows:

$$\begin{aligned} t_3 = & \frac{(B+1)(T-2)}{2B(T-1)}y + x + \frac{(B+1)(T^2-3T+4)}{4B(T-1)} \\ & + \frac{1}{2B(T-1)} \sum_{y' \neq y} \sum_{x'' \neq x} (max(y-1+y'-1, x'') \end{aligned}$$

Now we consider the case where the preceding operation is a retrieval. Similarly, we should separately derive the equations for $x'' = x$ and $x'' \neq x$. For the former case, the storage time can be calculated as follows:

$$t_4 = x + (y-1)$$

For the latter case, the storage time is the same as above if the last operation on tier y was a storage. If it was a retrieval, then the storage time is:

$$t_5 = \frac{1}{B-1} \sum_{(1 \leq x'' \leq B) \bigwedge x'' \neq x)} (max(y-1, x'')$$

Combining the above equations and weighting them with their probabilities, we have the storage time for the case where the preceding operation is a retrieval as follows:

$$\begin{aligned} t_6 = & \frac{B+1}{2B}y + x + \frac{B+1}{2B} \\ & + \frac{1}{2B(T-1)} \sum_{y' \neq y} \sum_{x'' \neq x} max(y-1, x'') \end{aligned}$$

Note that the probability that the preceding operation is a storage and that of a retrieval is identical, i.e., 0.5. We can combine t_3 and t_6 with their probabilities as weights, and derive the storage time for case b2 as follows:

$$\begin{aligned} t_s^{b2} = & \frac{(B+1)(2T-3)}{4B(T-1)}y + x + \frac{(B+1)(T^2-5T+6)}{4B(T-1)} \\ & + \frac{1}{4B(T-1)} \sum_{y' \neq y} \sum_{x'' \neq x} (max(y+y'-2, x'') + max(y-1, x'')) \end{aligned}$$

∎

Claim 4.4 *Retrieval time for case a, where* $(x, y) = (x', y')$ *is given by:*

$$t_r^a = x + y - 1$$

Proof: The preceding operation in this case must be storage so that the cell can be filled for this retrieval operation. Therefore, according to Property 4.1, we have $d_v = x$ and $d_h = y$. With Property 4.2, the operation is then represented as:

$$\begin{aligned} t_r^a(x,y) &= max(x,0) + max(0, y-1) \\ &= x + (y-1) \end{aligned}$$

■

Claim 4.5 *Retrieval time for case b1, where* $(x,y) \neq (x',y') \bigwedge (y = y')$ *is given by:*

$$t_r^{b1} = \frac{1}{2}x + y - 1 + \frac{1}{2}max(y-1, 2x) + \frac{1}{2(B-1)} \sum_{(1 \leqslant x' \leqslant B) \bigwedge (x' \neq x)} |x' - x|$$

Proof: The preceding operation is either a storage or a retrieval with same probability 0.5. From Property 4.1 and 4.2, if the preceding operation is a storage, t_r^{b1} is:

$$\begin{aligned} t_r^{b1}(x,y) &= \frac{1}{B-1} \sum_{(1 \leqslant x' \leqslant B) \bigwedge (x' \neq x)} |x' - x| + x + (y-1) \\ &= x + (y-1) + \frac{1}{B-1} \sum_{(1 \leqslant x' \leqslant B) \bigwedge (x' \neq x)} |x' - x| \end{aligned}$$

Else it is:

$$t_r^{b1}(x,y) = max(y-1, 2x) + (y-1)$$

Combine the above two equations and get their weighted average with same weight 0.5, we have:

$$t_r^{b1}(x,y) = \frac{1}{2}x + (y-1) + \frac{1}{2}max(y-1, 2x) + \frac{1}{2(B-1)} \sum_{(1 \leqslant x' \leqslant B) \bigwedge (x' \neq x)} |x' - x|$$

■

Claim 4.6 *Retrieval time for case b1, where* $(x,y) \neq (x',y') \bigwedge (y \neq y')$ *is given by:*

$$\begin{aligned} t_r^{b2} = {} & (y-1) + \frac{1}{2B}max(x, y-1) + \frac{B-1}{4B}max(2x, y-1) \\ & + \frac{1}{4B(T-1)} \sum_{(1 \leqslant y' \leqslant T) \bigwedge (y' \neq y)} 2max(x, |y'-y|) + (B-1)max(2x, |y'-y|) \\ & + \frac{1}{4B} \sum_{(1 \leqslant x'' \leqslant B) \bigwedge (x'' \neq x)} max(|x'' - x| + x, y' - 1) \\ & + \frac{1}{4B(T-1)} \sum_{(1 \leqslant y' \leqslant T) \bigwedge (y' \neq y)} \sum_{(1 \leqslant x'' \leqslant B) \bigwedge (x'' \neq x)} max(|x'' - x| + x, |y' - y|) \end{aligned}$$

Proof: In this case, the preceding operation is to a different tier. We first consider the case that the preceding operation was a storage, assuming that the last operation on tier y was applied to cell (x'', y). If $x'' = x$, then it must have been a storage with a probability of $1/B$. According to Property 4.1 and 4.2, the retrieval time for this subcase can be written as:

$$t_1 = \frac{1}{T-1} \sum_{(1 \leqslant y' \leqslant T) \bigwedge (y' \neq y)} max(x, |y' - y|) + (y-1)$$

If $x'' \neq x$ (with a probability of $(B-1)/B$), the last operation on tier y could be a storage or a retrieval with equal probability, i.e., 0.5. The retrieval time if it was a storage can be calculated as:

$$t_2 = \frac{1}{(B-1)(T-1)} \sum_{(1 \leqslant y' \leqslant T) \bigwedge (y' \neq y)} \sum_{(1 \leqslant x'' \leqslant B) \bigwedge (x'' \neq x)} max(|x'' - x| + x, |y' - y|) + (y-1)$$

whereas for the latter case, it can be calculated as:

$$t_3 = \frac{1}{T-1} \sum_{(1 \leqslant y' \leqslant T) \bigwedge (y' \neq y)} max(2x, |y' - y|) + (y-1)$$

Combining the above t_1, t_2 and t_3 and weighting them with their probabilities, we have the retrieval time for the case where the preceding operation is a storage as follows:

$$\begin{aligned} t_4 = (y-1) + \frac{1}{B(T-1)} & \sum_{(1 \leqslant y' \leqslant T) \bigwedge (y' \neq y)} max(x, |y' - y|) \\ \frac{1}{2B(T-1)} & \sum_{(1 \leqslant y' \leqslant T) \bigwedge (y' \neq y)} \sum_{(1 \leqslant x'' \leqslant B) \bigwedge (x'' \neq x)} max(|x'' - x| + x, |y' - y|) \\ + \frac{B-1}{2B(T-1)} & \sum_{(1 \leqslant y' \leqslant T) \bigwedge (y' \neq y)} max(2x, |y' - y|) \end{aligned}$$

Now we consider the case where the preceding operation is a retrieval. Similarly, we should separately derive the equations for $x'' = x$ and $x'' \neq x$. For the former case, the retrieval time can be calculated as follows:

$$t_5 = max(x, (y-1)) + (y-1)$$

For the latter case, the retrieval time is given as follows if the last operation on tier y was a storage.

$$t_6 = \frac{1}{B-1} \sum_{(1 \leq x'' \leq B) \bigwedge x'' \neq x)} max(|x'' - x| + x, (y-1)) + (y-1)$$

If it was a retrieval, then the retrieval time is:

$$t_7 = max(2x, (y-1)) + (y-1)$$

Combining the above t_5, t_6 and t_7 and weighting them with their probabilities, we have the retrieval time for the case where the preceding operation is a retrieval as follows:

$$\begin{aligned}
t_8 = {} & (y-1) + \frac{1}{B} max(x, (y-1)) \\
& + \frac{1}{2B} \sum_{(1 \le x'' \le B) \bigwedge x'' \neq x} max(|x''-x| + x, (y-1)) + (y-1) + \frac{B-1}{2B} max(2x, (y-1))
\end{aligned}$$

Note that the probability that the preceding operation is a storage and that of a retrieval is identical, i.e., 0.5. We can combine t_4 and t_8 with their probabilities as weights, and derive the retrieval time for case b2 as follows:

$$\begin{aligned}
t_r^{b2} = {} & (y-1) + \frac{1}{2B} max(x, y-1) + \frac{B-1}{4B} max(2x, y-1) \\
& + \frac{1}{4B(T-1)} \sum_{(1 \leqslant y' \leqslant T) \bigwedge (y' \neq y)} 2max(x, |y'-y|) + (B-1) max(2x, |y'-y|) \\
& + \frac{1}{4B} \sum_{(1 \leqslant x'' \leqslant B) \bigwedge (x'' \neq x)} max(|x''-x| + x, y'-1) \\
& + \frac{1}{4B(T-1)} \sum_{(1 \leqslant y' \leqslant T) \bigwedge (y' \neq y)} \sum_{(1 \leqslant x'' \leqslant B) \bigwedge (x'' \neq x)} max(|x''-x| + x, |y'-y|)
\end{aligned}$$

$\blacksquare$

Appendix C

Publications

(1) Chuanyu Chen, Shell-Ying Huang, Wen Jing Hsu, Ah Cheong Toh, and Chee Kit Loh. Platform-based AS/RS for container storage. *IEEE 2003 International Conference on Automation and Robotics*, Taipei, Taiwan.

(2) Chuanyu Chen, Shell-Ying Huang, Wen Jing Hsu, Ah Cheong Toh, Chee Kit Loh, Tiancheng Song, Liang Keon Yow, Ah Kiong Ong, and Chi Fei Chan. Simulation and optimization of container yard operations: a survey. *2nd International Conference on Port and Maritime R&D and Technology*, 2003, Singapore.

(3) Chuanyu Chen, Shell-Ying Huang, Wen Jing Hsu, Ah Cheong Toh, Chee Kit Loh, Tiancheng Song, Liang Keon Yow, Ah Kiong Ong, and Chi Fei Chan. A survey on simulation and optimization of container yard operations. *Singapore Maritime and Port Journal.* In Press.

(4) Ya-Hong Hu, Shell Ying Huang, Chuanyu Chen, Wen-Jing Hsu, Ah Cheong Toh, Chee Kit Loh, and Tiancheng Song. Travel time analysis of a new automated storage and retrieval system. *Computers & Operations Research*, In Press.

(5) Hu Yahong, Huang Shell Ying, Chen Chuanyu, Hsu Wen-Jing, Toh Ah Cheong, Loh Chee Kit and Song Tiancheng. A new automated storage and retrieval system and its travel time analysis. *IEEE Automation 2003*, September 12 14, 2003, Taiwan.

(6) Hu Yahong, Huang Shell Ying, Chen Chuanyu, Hsu Wen-Jing, Toh Ah Cheong, Loh Chee Kit and Song Tiancheng. Travel time analysis of a new automated storage and retrieval system. *IEEE Conference on Emerging Technologies and Factory Automation 2003*, Volume: 1, September 16-19, 2003, Portugal

(7) Hu Yahong, Huang Shell Ying, Chen Chuanyu and Hsu Wen-Jing. A New Automated Storage and Retrieval System and Its Travel Time Analysis. Submitted to Computers & Industrial Engineering. In review.

Bibliography and References

[AHBW01] J. Ashayeri, R.M. Heuts, M. Beekhof, and M.R. Wilhelm. On the determination of class-based storage assignment in an AS/RS having two I/O locations. *A discussion paper*, 2001.

[Bar94] Y. Bartal. A fast memoryless 2-server algorithm in Euclidean spaces. Unpublished manuscript, 1994.

[BDBK$^+$94] S. Ben-David, A. Borodin, R.M. Karp, G. Tardos, and A. Wigderson. On the power of randomization in online algorithms. *Algorithmica*, 11:2–14, 1994.

[Ber00] J.P. Van Den Berg. Simulation study of an automated storage/retrieval system. *International Journal of Production Research*, 38(6):1339–1356, 2000.

[BEY98] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis.* Cambridge University Press, 1998.

[BGR99] A.G. Bruzzone, P. Giribone, and R. Revetria. Operative requirements and advances for the new generation simulators in multimodal container terminals. *Proceedings of the 1999 Winter Simulation Conference*, 1:1243–1252, 1999.

[Boz84] Y.A. Bozer. Travel-time models for autmoated storage/retrieval systems. *IIE Transactions*, pages 329–338, 1984.

[CCH00] C.Y. Chen, S.L. Chao, and T.W. Hsieh. A time-space network model for the space resource allocation problem in container marine transportation. *Proceedings of the 17th Internationl Symposium on Mathematical Programming, Atlanta*, 2000.

[CD93] B. De Castilho and C.F. Daganzo. Handling strategies for import containers at marine terminals. *Transportation Research (B)*, 27(2):151–166, 1993.

[CDRS93] D. Coppersmith, P.G. Doyle, P. Raghavan, and M. Snir. Random walks on weighted graphs, and applications to online algorithms. *Journal of ACM*, 40:421–453, 1993.

[CE97] S.H. Chang and P.J. Egbelu. Relative pre-positioning of storage/retrieval machines in automated storage/retrieval systems to minimize maximum system response time. *IIE Transactions*, 29:303–312, 1997.

[CH99] C.Y. Chen and T.W. Hsieh. A time-space network model for the berth allocation problem. *19th IFIP TC7 Conference on System Modelling and Optimization, Cambridge, UK*, 1999.

[Che99] T. Chen. Yard operations in the container terminal - unproductive moves. *Maritime Policy and Management*, 26(1):27–38, 1999.

[CHH+03] C.Y. Chen, W.J. Hsu, S.Y. Huang, A.C. Toh, and C.K. Loh. Platform-based AS/RS for container storage. *IEEE International Conference on Robotics and Automation*, 2003.

[CKPV91] C. Chrobak, H. Karloff, T. Payne, and S. Vishwanathan. New results on server problems. *SIAM Journal on Discrete Mathematics*, 4(2):172–181, 1991.

[CL91a] C. Chrobak and L.L. Larmore. An new approach to the server problem. *SIAM Journal on Discrete Mathematics*, 4(3):323–328, 1991.

[CL91b] C. Chrobak and L.L. Larmore. An optimal online algorithm for k-servers on tress. *SIAM Journal on Computing*, 20(1):144–148, 1991.

[CLL02] R. Cheung, C.-L. Li, and W. Lin. Interblock crane deployment in container terminals. *Transportation Science*, 36(1):79–93, 2002.

[CLS95] C.S. Chen, S.M. Lee, and Q.S. Shen. An analytical model for the container loading problem. *European Journal of Operational Research*, 80(1):68–76, 1995.

[CLW+03] C.Zhang, J. Liu, Y. Wan, K.G. Murty, and R.J. Linn. Storage space allocation in container terminals. *Transportation Research (B)*, 37(10):883–903, 2003.

[CRM88] Y.G. Chuna, S.U. Randhawa, and E.D. Mcdowell. A simulation analysis for a transtainer-based container handling facility. *Computers and Industrial Engineering*, 14(2):113–125, 1988.

[CW97] D.T. Chang and U.P. Wen. The impact on rack configuration on the speed profile of the storage and retrieval machine. *IIE Transactions*, 29:525–531, 1997.

[CWL95] D.T. Chang, U.P. Wen, and J.T. Lin. The impact of acceleration/deceleration on travel time models for automated storage/retrieval system. *IIE Transaction*, 27(1):108–111, 1995.

[DB99] A.P. Davies and E.E. Bishoff. Weight distribution considerations in container loading. *European Journal of Operational Research*, 114:509–527, 1999.

[Egb91] P.J. Egbelu. Framework for dynamic positioning of storage/retrieval machines in an automated storage/retrieval system. *International Journal of Production Research*, 29(1):17–37, 1991.

[Els91] E.A. Elsayed. Ordering sequencing in automated storage/retrieval systems with due dates. *Progress in Material handling and Logistics*, 2:245–267, 1991.

[ER94] A. Eynan and M. Rosenblatt. Establishing zones in single-command class-based rectangular AS/RS. *IIE Transactions*, 26(1):38–46, 1994.

[EU89] E.A. Elsayed and O.I. Unal. Order batching algorithms and travel time estimation for automated storage/retrieval systems. *International Journal of Production Research*, 27(7):1097–1114, 1989.

[EW93] P.J. Egbelu and C.T. Wu. A comparison of dwell point rules in automated storage/retrieval system. *International Journal of Production Research*, 31(11):2515–2530, 1993.

[GGB93] A. Gunal, E.S. Grajo, and D. Blanck. Generalization of an as/rs model in SIMAN/CINEMA. *Proceedings of the 1989 Winter Simulation Conference*, pages 857–865, 1993.

[GHS77] S.C. Graves, W.H. Hausman, and L.B. Schwarz. Storage-retireval interleaving in automatic warehousing systems. *Management Science*, 23(9):935–945, 1977.

[GR69] M. Goetschalckx and H.D. Ratliff. Shared storage policies based on the duration of stay of unit loads. *Management Science*, 1990:1120–1132, 36(9).

[GRM98] L.M. Gambardella, A.E. Rizzoli, and M.Zaffalon. Simulation and planning of an intermodal container terminal. *Special Issue of SIMULATION on Harbour and Maritime Simulation*, 1998.

[Gro91] E. Grove. The harmonic online k-server algorithm is competitive. *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing*, pages 260–266, 1991.

[HL82] T.J. Hodgson and T.J. Lowe. Production lot sizing with material handling cost considerations. *IIE transactions*, 14(1):45–51, 1982.

[HL88] H. Hwang and M.K. Lee. Order batching algorithms for a man-on-board automated storage and retrieval system. *Engineering Costs and Production Economics*, 13:285–294, 1988.

[HL90] H. Hwang and S.B Lee. Travel-time models consider the operating characteristics of the storage and retrieval machine. *International Journal of Production Research*, 28(10):1779–1789, 1990.

[HMSW87] M.H. Han, L.F. McGinnis, J.S. Shieh, and J.A. White. On sequencing retrievals in an automated storage/retrieval system. *IIE Transactions*, 19(3):56–66, 1987.

[HSG76] W.H. Hausman, L.B. Schwarz, and S.C. Graves. Optimal storage assignment in automatic warehousing systems. *Management Science*, 22(6):629–638, 1976.

[HVJD99] J. Holguin-Veras and S. Jara-Diaz. Optimal pricing for priority service and space allocation in container ports. *Transportation Research (B)*, 33:81–106, 1999.

[IMP$^+$] M. Itmi, D. Morel, J.-P. Pecuchet, F. Serin, and L. Villefranche. Eco-problem solving for containers stacking, 1995.

[KB98] K.H. Kim and J.W. Bae. Re-marshaling export containers in port container terminals. *Computers and Industrial Engineering*, 35(3-4):655–658, 1998.

[Kim97] Kap Hwan Kim. Evaluation of the number of rehandles in container yards. *Computers and Industrial Engineering*, 32(4):701–711, 1997.

[KK96] H.B. Kim and K.H. Kim. A segregating space allocation for import containers in port container terminals. *Proceedings of the 20th International Conference on Computers and Industrial Engineering*, 2:1263–1266, 1996.

[KK97] K.Y. Kim and K.H. Kim. A routing algorithm for a single transfer crane to load export containers onto a containership. *Computers and Industrial Engineering*, 33(3-4):673–676, 1997.

[KK98] K.H. Kim and H.B. Kim. The optimal determination of the space requirement and the number of transfer cranes for import containers. *Computers and Industrial Engineering*, 35(3-4):427–430, 1998.

[KK99a] K.H. Kim and H.B. Kim. Segregating space allocation models for container inventories in port container terminals. *International Journal of Production Research*, 59(1-3):415–423, 1999.

[KK99b] K.Y. Kim and K.H. Kim. An optimal routing algorithm for a transfer crane in port container terminals. *Transportation Science*, 33(1):17–33, 1999.

[KK99c] K.Y. Kim and K.H. Kim. A routing algorithm for a single straddle carrier to load export containers onto a containership. *International Journal of Production Economics*, 59(1-3):425–433, 1999.

[KK02] K.H. Kim and H.B. Kim. The optimal sizing of the storage space and handling facilities for import containers. *Transportation Research (B)*, 36:821–825, 2002.

[KM88] A. Kaylan and D.J. Medeiros. Analysis of storage policies for miniload AS/RS. *Engineering Costs and Production Economics*, 13:311–318, 1988.

[KOSB99] S. Kul, N.E. Ozdemirel, C. Sepil, and Z. Bozkurt. Experimental investigation of shared storage assignment policies in automated storage/retrieval systems. *IIE Transactions*, 31:739–749, 1999.

[KP94] A. Keserla and B. Peters. An analysis of dual shuttle automated storage/retrieval systems. *Journal of Manufacturing Systems*, 13:424–434, 1994.

[KP95a] E. Koutsoupias and C. Papadimitriou. On the k-server conjecture. *Journal of the ACM*, 42(5):971–983, 1995.

[KP95b] P. Kouvelis and V. Papanicolaou. Expected travel-time and optimal boundary formulas for a two-class-based automated storage/retrieval system. *International Journal of Production Research*, 33(10):2889–2905, 1995.

[KP99] E. Kozan and P. Preston. Genetic algorithms to schedule container transfers at multimodal terminals. *International transactions in Operational Research*, 6(3):311–329, 1999.

[KP02] Kap Hwan Kim and Kang Tae Park. A note on a dynamic space-allocation method for outbound containers. *European Journal of Operational Research*, 2002.

[KPR00] Kap Hwan Kim, Young Man Park, and Kwang-Ryul Ryu. Deriving decision rules to locate export containers in container yards. *European Journal of Operation Research*, 124:89–101, 2000.

[LL94] K.K. Lai and K. Lam. A study of container yard equipment strategy in hong kong. *International Journal of Modelling and Simulation*, 14(3):134–138, 1994.

[LLW$^+$03] R. Linn, J. Liu, Y. Wan, C. Zhang, and K.G. Murty. Rubber tired gantry crane deployment for container yard operations. *Computers and Industrial Engineering*, 45:429–442, 2003.

[LS96] H.F. Lee and S.K. Schaefer. Sequencing methods for automated storage and retrieval systms with dedicated storage. *Conputers and Industrial Engineering*, 32(2):351–362, 1996.

[LTC99] Y.H. Lee, J.M.A. Tanchoco, and S.J. Chun. Performance estimation models for AS/RS with unequal sized cells. *Internation Journal of Production Research*, 37(18):4197–4216, 1999.

[Mal00] C.J. Malmborg. Interleaving models for the analysis of twin shttle automated storage and retrieval systems. *Internation Journal of Production Research*, 38(18):4599–4610, 2000.

[Mal01] C.J. Malmborg. Rule of thumb heuristics for configuring storage racks in automated storage and retrieval systems design. *International Journal of Production Research*, 39(3):511–527, 2001.

[Man97] M. Mansuri. Cycle-time computation, and dedicated storage assignment, for AS/R systems. *Computer and Industrial Engineering*, 33(1-2):307–310, 1997.

[MK01] G. Moon and G.P. Kim. Effects of relocation to AS/RS storage location policy with production quantity variation. *Computers and Industrial Engineering*, 40:1–13, 2001.

[MLR98] P. Montulet, A. Langevin, and D. Riopel. Minimizing the peak load: an alternate objective for dedicated storage. *International Journal of Production Research*, 36(5):1369–1385, 1998.

[MM97] R.D. Meller and A. Mungwattana. Multi-shuttle automated storage/retrieval systems. *IIE Transactions*, 29:925–938, 1997.

[MMS90] M.S. Manasse, L.A. McGeoch, and D.D. Sleator. Competitive algorithms for server problems. *Journal of Algorithms*, 11:208–230, 1990.

[MRP98] S. Mahajan, B.V. Rao, and B.A. Peters. A retrieval sequencing heuristic for miniload end-of-aisle automated storage/retrieval systems. *International Journal of Production Research*, 36(6):1715–1731, 1998.

[Mul89] D.J. Muller. AS/RS warehouse modelling. *Proceedings of the 1989 Winter Simulation Conference*, pages 802–810, 1989.

[Ng04] W.C. Ng. Crane scheduling in container yards with inter-crane interference. *European Journal of Operational Research*, in press, 2004.

[Par91] B.C. Park. *Analytical models and optimal strategies for automated storage/retrieval system operations.* PhD thesis, DGeorgia Institute of Technology, Atlanda GA, 1991.

[Par99] B.C. Park. Optimal dwell point policied for automated storage/retrieval systems with dedicated storage. *IIE Transactions*, 31:1011–1013, 1999.

[Par01a] B.C. Park. An optimal dwell point policy for automated storage/retrieval systems withu niformly distributed, rectangular racks. *International Journal of Production Research*, 39(7):1469–1480, 2001.

[Par01b] Byung Chun Park. An optimal dwell point policy for automated storage/retrieval system with uniformly distributed rectangular racks. *International Journal of Production Research*, 39:1469–1480, 2001.

[Psa98] H.N. Psaraftis. When a port calls... an operations researcher answers. *ORMS Today*, April, 1998.

[PSH96] B.A. Peters, J.S. Smith, and T.S. Hale. Closed form models for determining the optimal dwell point location in automated storage/retrieval systems. *International Journal of Production Research*, 34:1757–1771, 1996.

[PW99] C.H. Pan and C.H. Wang. A framework for the dual command cycle travel time model in automated warehousing system. *International Journal of Production Research*, 34:2117, 2099.

[RE89] M.J. Rosenblatt and A. Eynan. Deriving the optimal boundaries for class-based automatic storage/retrieval systems. *Management Science*, 35(12):1519–1524, 1989.

[Rou96] E.D. Roux. *Storage capacity for import containers at seaports.* PhD thesis, University of California, Berkeley, 1996.

[RS95] S.U. Randhawa and R. Shroff. Simulation-based design evaluation of unit load automated storage/retrieval systems. *Computers and Industrial Engineering*, 28(1):71–79, 1995.

[SB95] B.R. Sarker and P.S. Bahu. Travel time models in automated storage/retrieval systems: A critical review. *International Journal of Production Economics*, 40:173–184, 1995.

[Sch99] G. Scheithauer. Lp-based bounds for the container and multi-container loading problem. *International Transactions in Operational Research*, 6(2):199–213, 1999.

[SSLH91] B.R. Sarker, A. Sabapathy, A.M. Lal, and M.H. Han. Performance evaluation of a double shuttle automated storage and retrieval system. *Proceedings of Planning Control*, 2(3):207–213, 1991.

[Tak89] S. Takakuwa. Module modelling and economic optimization for lager-scale AS/RS. *Proceedings of the 1989 Winter Simulation Conference*, pages 795–801, 1989.

[Tak93] S. Takakuwa. Design and cost-effectivness analysis of large-scale AS/RS-AGV systems. *Proceedings of the 1989 Winter Simulation Conference*, pages 851–856, 1993.

[Tak94] S. Takakuwa. Precise modelling and analysis of large-scale AS/RS. *Proceedings of the 1989 Winter Simulation Conference*, pages 1001–1007, 1994.

[Tak96] S. Takakuwa. Efficient module-based modelling for a large-scale AS/RS-AGV system. *Proceedings of the 1989 Winter Simulation Conference*, pages 1141–1148, 1996.

[Tea03] HCTS Project Team. Development of a high capacity container terminal for mega-vessels. Project final report, Centre for Advanced Information Systems, School of Computer Engineering, Nanyang Technological University, 2003.

[Tho98] U.W. Thonemann. Optimal storage assignment policies for automated storage and retrieval systems with stochastic demands. *Management Science*, 44(1):142–148, 1998.

[TICD93] M. Taleb-Ibrahimi, B. De Castilho, and C.F. Daganzo. Storage space vs handling work in container terminals. *Transportation Research (B)*, 27:13–32, 1993.

[vdB02] J.P. van den Berg. Analytical expressions for the optimal dwell point in an automated storage/retrieval system. *International Journal of Production Economics*, 76:13–25, 2002.

[vdBG99] J.P. van den Berg and A.J.R.M. Gademann. Optimal routing in an automated storaeg/retrieval system. *IIE Transactions*, 31:407–415, 1999.

[Wat91] I. Watanabe. Characteristics and analysis method of efficiencies of container terminals: An approach to the optimal loading/unloading method. *Container Age*, pages 36–47, 1991.

[WCC01] U.P. Wen, D.T. Chang, and S.P. Chen. The impact of accelcration/deceleration on travel-time models in class-based automated S/R systems. *IIE Transactions*, 33:599–608, 2001.

[YC99] Won Young Yun and Yong Seok Choi. A simulation model for container terminal operation analysis using object-oriented approach. *International Journal of Production Economics*, 59:221–230, 1999.

[Zol82] H.A. Zollinger. Planning, evaluating and estimating storage systems. *Institute of Material Management Education First Annual Winter Seminar Series*, 10:1–10, 1982.

[Zol96] H.A. Zollinger. Expanded methodology to concept horizontal transportation problem. *Progress in material Handling Research*, Eidted by R.J. Graves et al.:651–663, 1996.

[ZWLL02] C. Zhang, Y. Wan, J. Liu, and R.J. Linn. Dynamic crane deployment in container storage yards. *Transportation Research Part B*, 36:537–555, 2002.