Comprehensive Methodology for the Design Configuration and Operational Control of Shuttle-based Storage and Retrieval Systems

by

Donghuang Li

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
May 7, 2022

Keywords: SBS/RS, Automated Warehouse,
Simulation, Queuing Analysis, Storage Assignment, Scheduling

Approved by

Jeffrey S. Smith, Chair, Professor of Industrial and Systems Engineering
Gregory Harris, Associate Professor of Industrial and Systems Engineering
Aleksander Vinel, Associate Professor of Industrial and Systems Engineering
Konstantinos Mykoniatis, Assistant Professor of Industrial and Systems Engineering

Abstract

During recent years, Autonomous Vehicle Storage and Retrieval Systems (AVS/RS) have been widely applied in distribution centers and production sites to meet the increasing demand for rapid and flexible large-scale warehousing activities. As the core subsystem of the bigger warehousing system, in AVS/RS horizontal devices (vehicles) and vertical devices (lifts) are applied for storage and retrieval operations to fulfill customer orders. The main topic of this study, Shuttle-based Storage and Retrieval System (SBS/RS), is one representative class in AVS/RS category. Recognizing the complex service dynamics due to the use of different types of S/R devices, both the configuration design problem and operational control problem need to be studied in order to improve efficient, sustainable and robust performance of the system. We identified critical decisions and factors of SBS/RS-based warehousing systems and studied their complexities and correlations, and proposed a comprehensive and consistent methodology for both the design configuration and the operational control practices.

In the first step, an animated, data-driven and data-generated simulation model is developed to support the development of both the design and configuration approach and the operational control strategy of SBS/RS-based warehouse systems. The model enables detailed analysis of different technology options including tier-captive and tier-to-tier configurations, multi-deep rack designs, multi-capacity lifts, etc., and provides visualized tracking of accurately simulated service processes of the S/R devices and performance evaluation under configurable demand scenarios.

In the second step, the research focuses on the conceptual design problem in which alternative SBS/RS designs and configurations are evaluated. A three-stage design methodology based on queuing analysis is proposed to provide rapid evaluation and screening for large number of alternative design options. The methodology considers not only design configuration parameters including number of aisles, tiers, columns, and shuttles, but also different technology options as well as control policies to improve estimation quality.

In the third step, we focus on the control aspect which involves the development of various operational control approaches for storage assignment and device scheduling. Mathematical

programming techniques and dynamic dispatching approaches are explored to provide in-depth analysis of the control decisions. An operational control strategy framework that systematically integrates the control policies is developed and illustrated in detailed charts and pseudocode for practical implementation.

The research work used in this dissertation is part of a research and development project of an SBS/RS product by *Damon Group* (http://www.damon-group.com/), a global logistics solution provider, which is also the sponsor of this research work. Thanks to the practical data and professional insights provided by our sponsor, this research work is examined and validated all the way along its development.

Acknowledgements

I would like to first express sincere gratitude to Dr. Jeffrey Smith, my advisor, for his guidance, support, encouragement and patience throughout all the years of my PhD study. I would also like to appreciate Dr. Yingde Li and Dr. Yuan Sun, for their great support to this research. I also want to thank my committee members, Dr. Aleksander Vinel, Dr. Gregory Harris, and Dr. Konstantinos Mykoniatis, for their insights and suggestions along my dissertation process. In addition, I would like to thank my colleagues and friends: Dr. Gang Hao, Dr. Mohammad Ansari, and Mr. Zequn (Sylar) Liu, for all kinds of help. Finally, I would like to thank my family members – my parents and my wife – for their love and accompany that supported me overcoming all the thorns.

TABLE OF CONTENTS

Chapter 1 Introduction

1.1 System Overview

With the rise of e-commerce and advancement of Industry 4.0 technologies, automated storage and retrieval system (AS/RS) technologies are evolving rapidly to help supply chain enterprises achieving higher performance, lower costs, and better responsiveness to customers and supply chain partners. During the past decade, a preeminent AS/RS innovation, autonomous vehicle storage and retrieval system (AVS/RS) technology, has been increasingly introduced to the market and widely applied in automated production and warehouse control. An AVS/RS typically uses separate vertical machines (lifts/elevators) and horizontal machines (vehicles/shuttles) for items' storage and retrieval operations to and from locations in storage racks. Compared with traditional crane-based automated storage and retrieval systems, AVS/RSs provide higher transaction rate and better responsiveness to orders, and also improve overall system flexibility in accommodating stochastic and seasonal demands. The technology has proven successful especially for use with relatively lightweight loads, large product assortments, and low order sizes. Various system designs based on AVS/RS technology are developed. The main topic of this study, Shuttle-based Storage and Retrieval System (SBS/RS) (also called Multi-shuttle Storage and Retrieval System), is one representative subset of AVS/RS category (see Figure 1.1).

In general, activities in automated warehouses are designed to fulfill orders from customers or partners from the supply chain. Fulfilling an order typically consists of picking up certain quantities of specific SKUs (stock keeping units) from storage locations (slots), and packaging into containers, possibly with specific sequence/sorting requirements. This order-fulfillment operation is sometimes called "pick-up". Besides pick-up operations, replenishment operations of goods, containers, etc. are another important aspect to support the daily functions of the warehouse system.

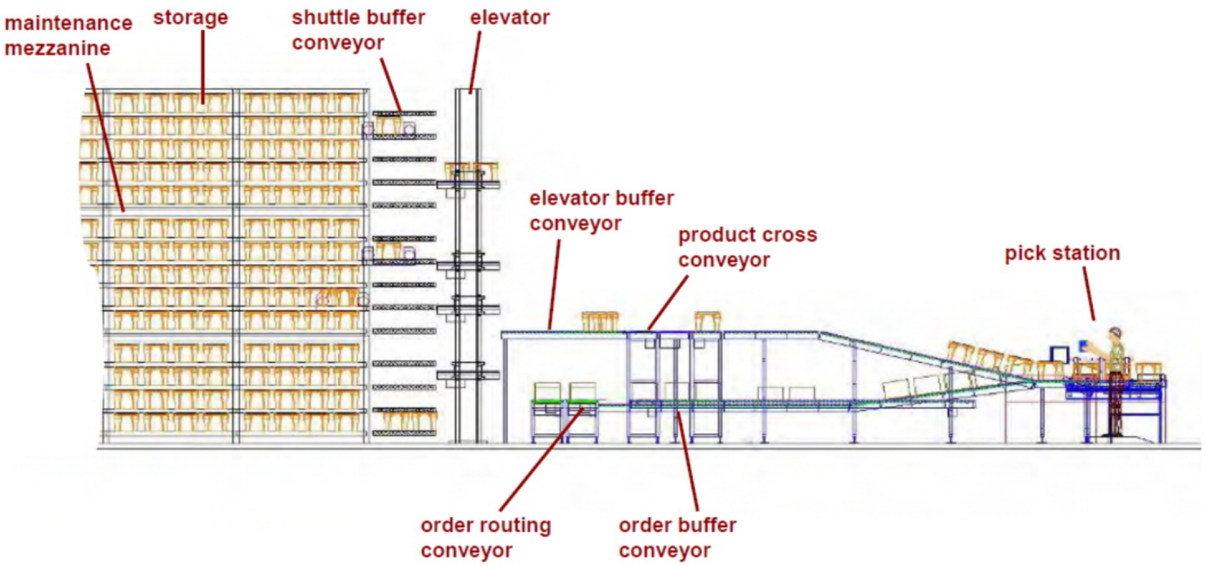

Figure 1.1 Illustration of a Shuttle-based Storage and Retrieval System (Dematic Multishuttle 2 Whitepaper, 2013)

An SBS/RS applies a number of shuttles (vehicles) and two non-passing lifts – a storage lift and a retrieval lift – for each aisle. In traditional SBS/RS, each tier is deployed with a dedicated shuttle (tier-captive), thus the number of shuttles is no longer configurable once the rack dimensions are determined. Such designs are effective in situations that the demand scenarios are less uncertain. Another type of SBS/RS is called tier-to-tier SBS/RS, where shuttles are transferred between tiers when needed by an additional aisle-captive shuttle lift. In this research, the design configuration methodology and operational control strategy for both tier-captive and tier-to-tier SBS/RS are studied.

Just like all other types of automated warehouse systems, the operations in both types of SBS/RS-based warehouse system can be described as: 1) storage operations, by which specific SKUs stored in container units (totes, bins, etc.) are delivered from system input terminals to specific storage locations (slots) in the rack; and 2) retrieval operations, by which specific SKUs stored in container units are accessed from specific slots in the rack and then delivered to system output terminals. However, unlike its traditional counterparts like crane-based automated storage and retrieval system (CBS/RS) in which a single automated mechanism takes care of all aspects of storage and retrieval operations, each S/R device in an SBS/RS aisle is only responsible for either the vertical parts (the lifts) or the horizontal parts (the shuttles) in storage and retrieval

services. The horizontal services are performed in parallel by multiple shuttles and are relatively decoupled from the vertical services, thus SBS/RS can potentially be more efficient and economical than many traditional S/R systems in high-speed demand environments – if designed and controlled properly. Moreover, in tier-to-tier SBS/RS the number of shuttles in each aisle is configurable, thus provides more flexibility in accommodating demand seasonality and reducing operational costs.

A typical SBS/RS-based warehouse system consists of multiple SBS/RS aisles that interface with the encompassing systems like the work (pick) stations and conveyor network (see Figure 1.1). The aisles, workstations and conveyor network are subsystems of the larger warehouse system that need to be designed and configured with a holistic methodology so that to cost-effectively fulfill the changing demands and controlled and coordinated in the operations so that to ensure efficient and sustainable performance of the storage and retrieval services. The design and configuration of SBS/RS requires precise and efficient performance estimation approaches for evaluating large numbers of design/configuration options. The development of operational control strategy involves clear identification of control decisions (storage assignment, task scheduling, etc.), and evaluation, integration and fine-tuning of control policies for different decisions.

1.2 System Workflows

Figure 1.2 shows how the devices within an SBS/RS aisle serve storage tasks and retrieval tasks. The aisle illustrated here is 2-deep, which means there are two storage rows in the z-axis on both sides of the aisle. Shuttles (autonomous vehicles) are responsible for horizontal movement for both storage tasks and retrieval tasks. A shuttle travels on rails on each tier, accesses storage locations of both depths using its arms, and each time carries no more than one tote. Two non-passing tote lifts, namely the storage lift and retrieval lift, are elevators responsible for vertical movement for storage tasks and retrieval tasks, respectively. Each tote lift is a continuous elevator so that it may carry more than one tote within a tour. Moreover, in a tier-to-tier aisle the total number of shuttles is usually less than number of tiers, and there is a dedicated shuttle lift responsible for transferring shuttles between tiers when needed. In a tier-captive aisle, each tier has a captive shuttle and there is no shuttle lift.

1.2.1 Shuttle Operations

The shuttle and shuttle lift operations are further explained in Figure 1.3. The input/output buffer conveyors on the left-most side of each tier are temporary stacking locations between shuttle (horizontal) services and tote lift (vertical) services. In a retrieval task, the service shuttle (shuttle.1) moves from its current column to the column of the tote's slot, loads the tote, then moves to the output buffer on the tier, and finally unloads the tote to the buffer. The shuttle is then released, and the tote is waiting for the next phase of retrieval service provided by the retrieval lift. In a storage task, the tote to be stored is delivered to the input buffer conveyor through the first phase of storage service provided by the storage lift. The service shuttle (shuttle.2) then moves from its current column to the input buffer, loads the tote, then moves to the column of the tote's target slot, and finally unloads the tote to slot. The shuttle is then released. In addition, for 2-deep aisles, a retrieval task from a 2-deep slot could be blocked by a tote stored at the slot's neighboring 1-deep slot – in such cases, relocation of the blocker tote needs to be performed by the shuttle.

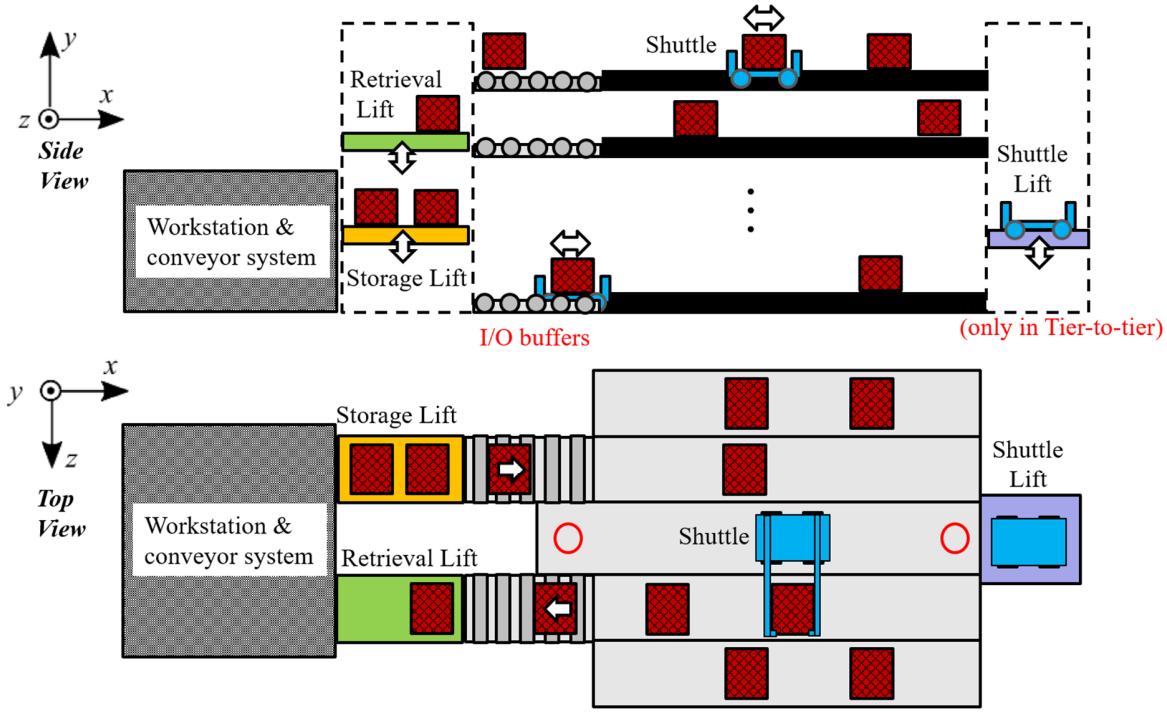

Figure 1.2 Operations within a single aisle of a tier-to-tier SBS/RS

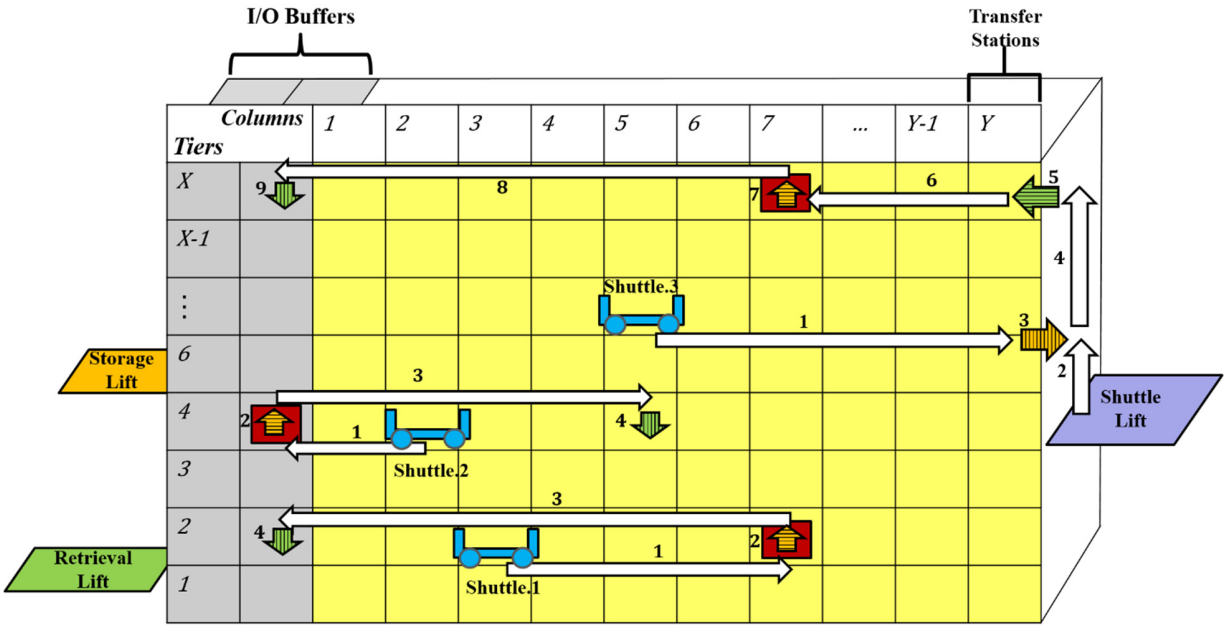

Figure 1.3 Shuttle Operations Illustration of a tier-to-tier SBS/RS

In cases where there is no available shuttle to serve a storage task to a target tier or retrieval task from a target tier, shuttle tier-transfer operations must be conducted. First, a shuttle (shuttle.3) from another tier is selected and moves from its current column to transfer station at the right-most side of the tier. Then, the shuttle lift moves from its current tier to the shuttle's tier, load the shuttle, then carries the shuttle to the target tier of the task, and finally unload the shuttle at the transfer station on the target tier. The shuttle lift is then released, and the shuttle proceed to serve the task.

1.2.2 Tote Lift Operations

Operations of the storage lift and retrieval lift are illustrated in Figure 1.4. The tote lifts can be either 1-capacity or multi-capacity depending on the technology applied – for the latter, the lifts can carry more than one tote at a time in each tour. In a storage task, the storage lift moves from its current tier to I/O floor to load the totes delivered by encompassing systems (e.g., conveyor system which delivers storage totes to the SBS/RS). Next, the lift moves to the target tiers of each storage totes, and finally unload the tote to the target tier's input buffer. In a retrieval task, the retrieval lift moves from its current tier to the tier of each retrieval tote and loads it, then moves to the I/O floor and finally unloads the tote.

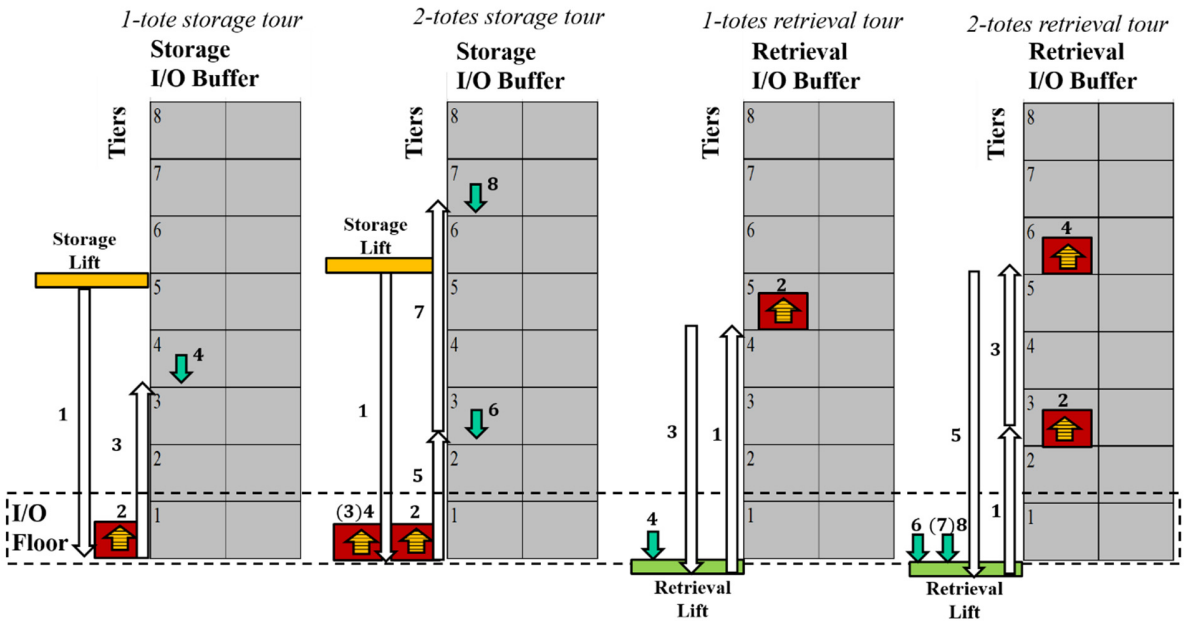

Figure 1.4 Tote Lift Operations Illustration of a tier-to-tier SBS/RS

1.3 Design and Control Decision Makings in SBS/RS-based Warehouses

1.3.1 Indicators for Evaluating Design and Control Decisions

For an AS/R System, critical performance indicators include travel time per request, number of requests handled per time period, total time required to handle a certain number of requests, waiting times of cranes of the AS/RS, waiting times of products to be stored/retrieved, and number of requests waiting to be stored/retrieved (Roodbergen and Vis, 2009). In this research, the major indicators of SBS/RS based warehouse system are summarized as:

Costs, which include the building costs of the rack aisles and aisle-captive lifts, the deployment costs of the shuttles, and the operational costs due to control strategies.

Throughput of the system is defined by the maximum number of customer orders the system can fulfill per unit time. The system becomes unstable when its throughput cannot meet the target demand environment.

Responsiveness of the system, which is measured by the cycle time of storage/retrieval requests, consist of both the service times of the S/R devices and the waiting times for device services.

The total required storage capacity is usually determined by the demand patterns of the expected application environment. In order to meet the storage capacity requirement, large numbers of design and technology options involving aisle numbers, rack dimensions and depths, tier-captive/tier-to-tier configurations, etc. are up for selection, while from which only one best design will be selected to fulfill the costs, throughput and responsiveness goals of the demand requirements and constraints of the application environments. In addition, the design of the SBS/RS usually must be considered together with the design of the encompassing systems and subject to the overall distribution/manufacturing processes.

Although the system's throughput and responsiveness performance primarily depend on system design and configuration, there is always room for further improvement through wise application of operational control approaches. Those approaches include storage assignment policies, device scheduling policies, and order dispatching policies, etc., each representing one aspect of control decisions. Those control approaches are expected to be integrated systematically and respond dynamically to demand changes so that to ensure efficient, sustainable, and robust system performance.

1.3.2 Design and Configuration Decisions

Each SBS/RS aisle has multi-tier storage racks on both sides and will be configured with its own set of S/R devices. These devices include one storage lift and one retrieval lift that handle the vertical inbound and outbound movements of totes to/from the SBS/RS, multiple shuttles that move horizontally in the aisle direction, and (in tier-to-tier configurations) one shuttle lift that transports shuttles between tiers. During the conceptual design phase, design parameters of the SBS/RS are determined. At first, the total required storage capacity is determined based on estimation of the future demands of the application environment, and decision is made between the tier-captive and tier-to-tier technology options. Tier-to-tier aisles are usually more expensive in initial investments due to the additional shuttle lift and the shuttles designed for tier-transfer mechanisms but allows the decision-maker to flexibly reconfigure shuttle deployment facing future demand changes so that to reduce operational costs. Then, based on the storage capacity

requirements and slot dimension requirements, the number of aisles, as well as the number of tiers on the y-axis (thus the rack height) and the number of columns/bays on the x-axis (thus the rack length) each aisle, need to be determined. In addition, the rack depth on the z-axis needs to be decided for each aisle: comparing to 1-deep rack designs, 2-deep rack designs provide higher storage density but usually longer average service time due to the required relocation efforts. The layout of the warehouse space is the main constraint to those decisions, while the locations of encompassing systems and patterns of their processes could also play significant roles. Usually, each aisle has identical numbers of tiers and columns, however irregular aisle designs may be applied depending on the requirements of the application environment.

The incomplete knowledge of the demand scenario – not only the randomness of the demand in terms of throughput and responsiveness, but also the uncertainties with demand patterns, order structures, and SKU-level interdependencies – further complicate the design and configuration problem. Tier-to-tier SBS/RS's flexibility in changing shuttle quantities allows better response to demand changes. However, sensitivity analysis involving various demand forecasts is still necessary, in which shuttle deployment decisions need to be evaluated together with operational control strategies under different scenarios. Because the design parameters and decisions may yield large search spaces of design candidates, a precise and efficient analytical methodology is necessary for candidates' initial evaluation and screening.

1.3.3 Operational Control Decisions

To fully utilize the S/R devices and improve service performance, different control approaches need to be developed and evaluated by simulation or analytical techniques, integrated systematically in a larger control strategy framework, and response dynamically to incoming demands and system states. Based on our literature review (Chapter 2) and communications with our industry partners, we define three major types of operational control policies for SBS/RS:

1. *Order Dispatching*, which determines how demand orders for SKUs are decomposed into tasks, as well as the timing of releasing those tasks to different S/R devices. Depending on the application environment, additional complexities like order due date requirements and pick-up precedence constraints may be introduced. Failure in dispatching orders may lead to long waiting times and poor utilization of S/R devices.

2. *Storage Assignment*, which determines the storage positions for each incoming storage tote. A storage assignment approach may be realized through either static rules based on SKU characteristics, or dynamic policies based on the current device and inventory states, or a combination of both aspects. In 2-deep aisles, relocation decisions can also be viewed as variations of storage assignment decisions. Storage assignment/relocation decisions not only affect the current storage/relocation services but also the future retrieval services. Good storage assignment approaches are designed to reduce unnecessary device movements and improve the system's long-term efficiency in a robust way.
3. *Device Scheduling*, which determines the sequence or priority that each S/R device uses to serve existing tasks. For tote lifts and shuttles, each device can be viewed as operating following its own task schedule determined by the scheduling approaches. Design options and constraints involving rack depths, tote lift capacities, I/O buffer capacities, etc. bring additional complexities for the scheduling approaches. Moreover, for tier-to-tier systems two-degree decisions are made for tier-transfer services because both the target tiers and the shuttle allocation need to be selected. Good scheduling approaches are expected to create device schedules that improve the overall system performance, considering the direct and indirect interactions between the horizontal and vertical service processes, and adaptive to stochastic and changing demands.

Generally speaking, order dispatching is more of a managerial-level decision (e.g., at the Warehouse Management System level) and subject to the application environment, the device scheduling decisions are more execution-level decisions (e.g. at the Warehouse Execution System level), and the storage assignment decisions are somewhere in between. In this research, the control strategy development mainly focuses on the latter two types of control decisions. Other control aspects include dwell policies which determine each S/R device's action when it completed a service, entrance control of buffer conveyors, emergent task rules, etc. It is notable that, the control aspects mentioned above are correlated in implementation, while also subject to system design and demand scenario characteristics. Thus, the control approaches may need to be customized to demand patterns (e.g., SKU assortment) and updated based on the demand changes (e.g., SKU seasonality).

1.3.4 Interdependency of the Decisions

The control decisions are highly correlated, while also largely affected by demand patterns as well as system design and configuration decisions. Recognizing the multi-stage nature of SBS/RS processes, we identify the critical design and control decision factors and illustrate the interdependencies between those factors in Figure 1.5, where important dependencies among those are denoted and highlighted.

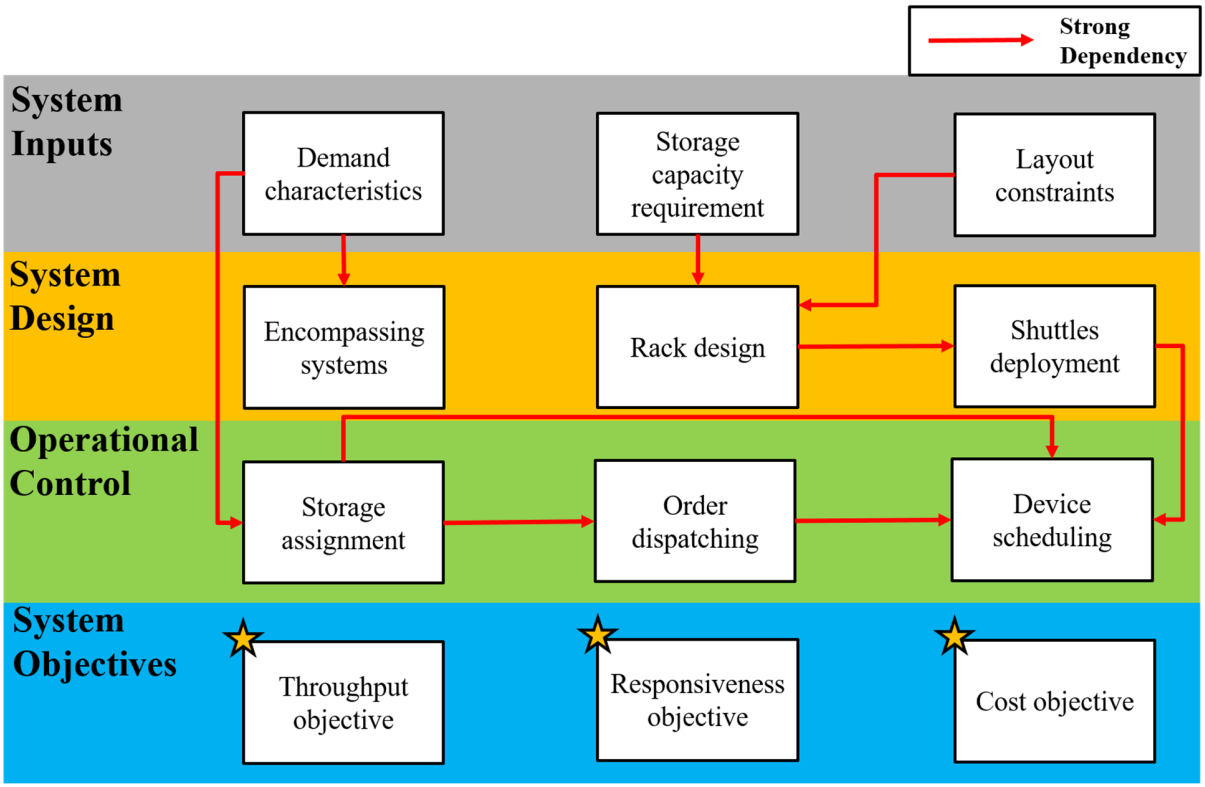

Figure 1.5 Key factors and decision-makings of SBS/RS-based warehouse design configuration and control strategy development

1.4 Research Methodology

The design configuration and operational control of SBS/RS are two closely related aspects, and both are highly dependent on the demand characteristics of the application environment. Our study aims to provide a comprehensive set of approaches from the system conceptual design phase to operational control practices to systematically improve the overall performance in different

application areas and under stochastic and changing demands. The objectives of this study can be concluded as follows:

1) Simulation Model Design and Development

In Chapter 3, an animated, data-generated and data-driven simulation model is built to support development of both the design and configuration methodology and operational control strategy of SBS/RS-based warehouse systems. The simulation model enables detailed analysis of different system configurations and technology options including tier-captive and tier-to-tier, multi-deep rack designs, multi-capacity lifts, etc., and provides visualized tracking of accurately simulated service processes of the S/R devices and performance evaluation under configurable demand scenarios in forms of stochastic task arrival process. In the following chapters, the development of the conceptual design methodology and the operational control strategy are highly dependent on the Monte Carlo experiments based on the simulation model. From the design perspective, the simulation model cross-validates the analytical model to facilitate the design methodology development. From the control perspective, various aspects of operational control strategy can be customized, evaluated, and fine-tuned to accommodate to different demand scenario assumptions, so that to facilitate the control strategy development.

2) System Design System Design, Configuration and Performance Analysis

In Chapter 4, a general analytical approach is established to support the SBS/RS conceptual design. A comprehensive travel time model based on queuing network analysis is developed to capture critical aspects from the system design and operational control perspectives. Based on the travel time model, a precise and efficient three-stage iterative analytical approach is proposed. In this approach, key parameters in SBS/RS design and configuration are identified, and critical variables describing stochastic demand scenarios are abstracted and integrated to the mathematical analysis. Design candidates – described by design parameters including numbers of aisles, tiers, columns, rack depth, and number of shuttles – are evaluated and screened, in order to find the best design for the given application environment described by demand variables including distributions of inter-arrival times of storage tasks and retrieval tasks as well as the inventory level (rack utilization) assumptions. System performance under evaluation is indicated by system throughput and task responsiveness (cycle times). Then, based on both the design parameters and demand/operational variables defined above, a queuing network model is developed to describe

the system and estimate performance. Various design options including tier-captive configurations and tier-to-tier configurations, multi-deep racks, and multi-unit tote lifts are accommodated in the model. For all these design options, work patterns of different types of S/R devices and their correlations are analyzed under a general framework.

One important observation from this research is that the expected performance of operational control also needs to be estimated during the conceptual design phase. This is due to the process complexity caused by both the demand pattern and the S/R device operations. The in-depth attempts integrating demand-level and operational-level factors in configuration design phase is one important novelty this research is expected to bring. Finally, the travel time model is validated by Monte-Carlo experiments with the simulation model. The validation results for both throughput and responsiveness estimates are satisfactory for both tier-captive and tier-to-tier configurations.

Through systematically and rapidly screening alternative system configurations with sufficient estimation preciseness, our analytical approach narrows down the search scope for detailed evaluation and further operational control strategy development. Thus, the proposed design and configuration methodology is expected to contribute to the body of knowledge of the automated warehouse research and provide useful and significant insights for industrial practitioners.

3) Operational Control Strategy Development

In Chapter 5, operational control strategies are studied aiming at optimizing the throughput and responsiveness performance of SBS/RS. Two types of control policies – storage assignment and device scheduling – are further explored. Both the short-term effects and long-term effects of the control policies are be considered.

Both mathematical programming (MP) techniques and dynamic dispatching (DD) approaches are explored to provide in-depth analysis of the control decisions and improve the overall performance of the control policies. The control strategy development is an incremental procedure starting from simple assumptions for system details and demand complexity. The control decisions are abstracted and formulated in MP optimization problems and augmented step-by-step along with the assumptions. DD approaches are continuously developed and evaluated by

studying the benchmark MP solutions and systematically and fine-tuned through simulation experiments. Storage assignment policies that accommodate different demand assumptions, and device scheduling policies that adapt tier-captive and tier-to-tier systems, are proposed.

The evaluation and validation of the control approaches are performed continuously using simulation experiments. We observe from simulation experiments that the proposed control strategy is computationally efficient, adaptive to various system design/configurations and demand scenario assumptions and improves the system performance significantly. Finally, an operational control strategy framework that systematically integrates the control policies is developed and illustrated in detailed charts and pseudocode for practical implementation. In addition, the time estimates from and updated by the control algorithms are expected to provide useful information for coordinating the encompassing subsystems in the larger warehouse system.

1.5 Summary

Due to the variety, stochasticity, complexity, and timing requirements brought by the rising e-commerce, coherent and systematic decision-making methodologies concerning both design and control aspects of automated warehousing systems are needed to improve the overall performance of warehousing systems. By using separate vertical and horizontal robotic S/R devices, Shuttle-based Storage and Retrieval System (SBS/RS) technology shows great potential in improving warehousing system efficiency. In addition, the flexibility of shuttle deployment decision enables better response to varying demands.

To improve the throughput and responsiveness performance, cost-effectiveness, and robustness of SBS/RS-based warehouse, a comprehensive methodology that facilitates system design configuration and control strategy integration in a valid, systematic, and consistent manner is proposed. By exploring research approaches involving analytical modeling, simulation modeling, mathematical programming, dynamic dispatching, and statistical analysis, etc., we expect our work to provide useful insights to industrial practitioners and contribute to the body of knowledge as well.

Chapter 2 Literature Review

2.1 Automated Storage and Retrieval System (AS/RS)

Warehouses, as well as production and distribution centers worldwide, have been applying Automated Storage and Retrieval System technologies since 1950s (Roodbergen and Vis, 2009). An Automated Storage and Retrieval System (AS/RS) is a combination of equipment and controls that handle, store, and retrieve materials as needed with precision, accuracy, and speed under a defined degree of automation (Material Handling Industry, 2020). Most types of AS/RSs consist two parts: the racks in which products/materials are stored, and the automated handling devices that execute storage and retrieval operations from/to encompassing systems – for example, picker workstations or production systems. AS/RSs are fully automated goods-to-person order-picking systems. Compared to their non-automated counterparts, AS/RSs provide lower labor costs, higher space utilization, better accuracy, and lower error rates.

In a recent survey research by Boysen et al. (2019), the authors pointed out four important aspects of new generation warehousing requirements in e-commerce area: **small orders**, **large assortment**, **tight delivery schedules**, and **varying workloads**. Warehousing operations, especially order-picking, are labor-intensive material handling operations. For more than half century, computer-directed AS/RS technologies have been proven particularly successful in large-scale warehousing applications. On the other hand, facing increasing challenges for dynamic demands, cost reduction, and throughput and responsiveness requirements, AS/RS technologies have been evolving into various branches, and each branch led to lots of research into all aspects on system design and configuration methodologies as well as operational control strategies.

The earliest type of AS/RS, Crane-based AS/RS, uses automated cranes with a load/unload shuttle as the only mechanism to access slots in rectangular racks and perform pick up or drop off operations. The storage racks are divided into aisles, and each aisle is installed with one crane (*aisle-captive*). The crane travel in such AS/RS follows Chebyshev pattern (travels on horizontal and vertical directions simultaneously), and all waiting storage/retrieval tasks cannot be started until the crane is released from its current task(s). A typical AS/RS is illustrated in Figure 2.1.

2.1.1 Design and Configuration of AS/RS

As the application environment of an AS/RS is typically highly dynamic and turbulent, the operational demand for the AS/RS is usually subject to stochasticity, uncertainty, and seasonality. To improve system performance and reduce investment costs, important design and configuration decision makings need to be made during the initial system conceptualization phase. In order to fulfill system requirements characterized by storage capacity, throughput, response time, etc., designers must decide the numbers of aisles, tiers (levels), columns, depth, and locations of input/output stations and so on for the AS/RS. Additional decisions may be made for different AS/RS variations.

Analytical modeling based on travel time analysis, as well as simulation modeling, are the two major categories of approaches to evaluate AS/RS performance and thus guide system design and configuration. In early research by Hausman et al. (1976) and Graves et al. (1977), both continuous representations and discrete procedures are developed to establish AS/RS travel-time models which provide good approximation under specific assumptions. Bozer and White (1984) developed travel time models for AS/R machines and considered alternative I/O locations. A review research focusing on AS/RS travel-time models is conducted by Sarker and Babu (1995). Malmborg (2001) proposed a rule of thumb heuristics to guide design and configuration of crane-based AS/RS. In a research by Gagliardi et al. (2012), various models for unit-load AS/RS – including both dynamic simulation-based ones and steady-state travel-time-based ones – are reviewed. The authors pointed out the limitations of analytical models due to the strict assumptions required, while also pointed out the limitations of simulation models as they are hardly generalizable.

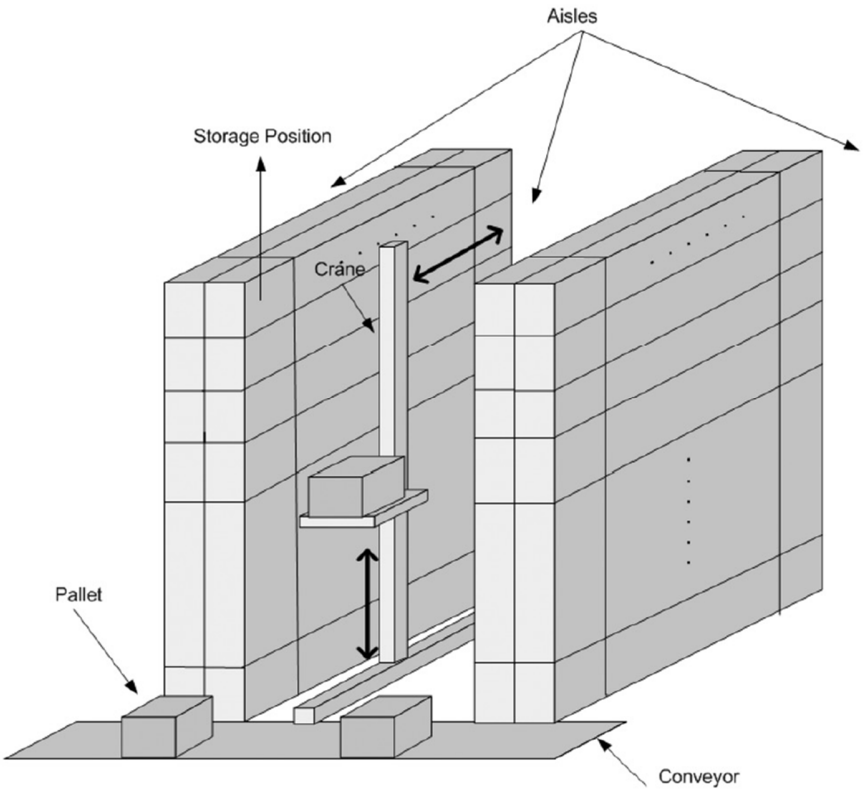

Figure 2.1 A typical AS/RS (Heragu et al., 2011)

2.1.2 Operational Control of AS/RS

A large number of research works have been conducted to develop AS/RS (Automated Storage and Retrieval System) operational control strategies to study control problems involving the storage assignment of products, the interleaving and dwell-point positioning of the S/R machine, the sequencing of tasks, and the batching of orders, etc. Simulation modeling techniques are widely applied in those works to evaluate the performance of the control strategies.

Hausman et al. (1976) compared the operating performance of four storage assignment rules – random storage, closest-open-location storage, class-based storage, and full turnover-based storage. Their work is also one of the first to consider request sequencing. Graves et al. (1977) compared the operating performance of alternative storage assignment / interleaving policies. Mandatory interleaving rules including First-Come-First-Serve (FCFS) and class-based prioritization rules are studied. Schwarz et al. (1978) used simulation to evaluate AS/RS storage assignment rules and interleaving rules under stochastic conditions. Bozer and White (1984)

examined various dwell point policies. Han et al. (1987) studied retrieval sequencing in AS/RS to improve system throughput, in which both analytical modeling and simulation technique is applied to evaluate sequencing rules. Two approaches for sequencing storage and retrieval requests, wave sequencing and dynamic sequencing, are proposed. Linn and Wysk (1987) developed a simulation model to analyze different control procedures of an AS/RS, considering effects of different product mix and seasonal demand trends. Eben-Chaime (1992) suggested a dynamic nearest-neighbor heuristic which outperformed the block-sequencing strategy in non-deterministic environments. In the review research by Sarker and Babu (1995), AS/RS travel time models for different command-cycle operations are developed for both single-shuttle and double-shuttle AS/R machines, and impacts of retrieval sequencing and order batching are discussed. An optimal AS/RS routing algorithm under dedicated storage policy is proposed by Gademann (1999). A set of storage and retrieval requests is assumed as given beforehand and no new requests come in during operation (block sequencing). The author formulated the sequencing problem as a Transportation Problem which can be solved in polynomial time. Van Den Berg and Gademann (2000) conducted a simulation study to evaluate for types of policies and rules: storage assignment, request selection, open location selection, and urgency rules. Boysen and Stephan (2016) presented a novel classification scheme for single-crane scheduling problem, in which varieties of AS/RS layout, order characteristics and objective function are included in their classification. Order characteristics are defined in terms of kind of requests, information availability, release dates, deadline, storage positions, and precedence relations.

2.1.3 Variations of AS/RS

Ever since the introduction of the first aisle-captive crane-based AS/RS, continuous innovations based on the original design have been invented to better meet requirements under different circumstances: some enabled the cranes moving across aisles (*moveable-aisle AS/RS*), thus reduced costs as fewer cranes are needed; some increased capacity of the L/U shuttle on crane (*multi-shuttle crane*) and proposed specific scheduling methods; some developed rotating racks (*carousels*) to reduce task cycles; some increased storage depth on z-axis (*double-deep rack*) to improve storage density. Each successful innovation further contributes to the body of knowledge and benefits the industries. Roodbergen and Vis (2009) conducted a comprehensive review of

AS/RS classifications and provided good insights on both system design aspects and operational control aspects (Figure 2.2).

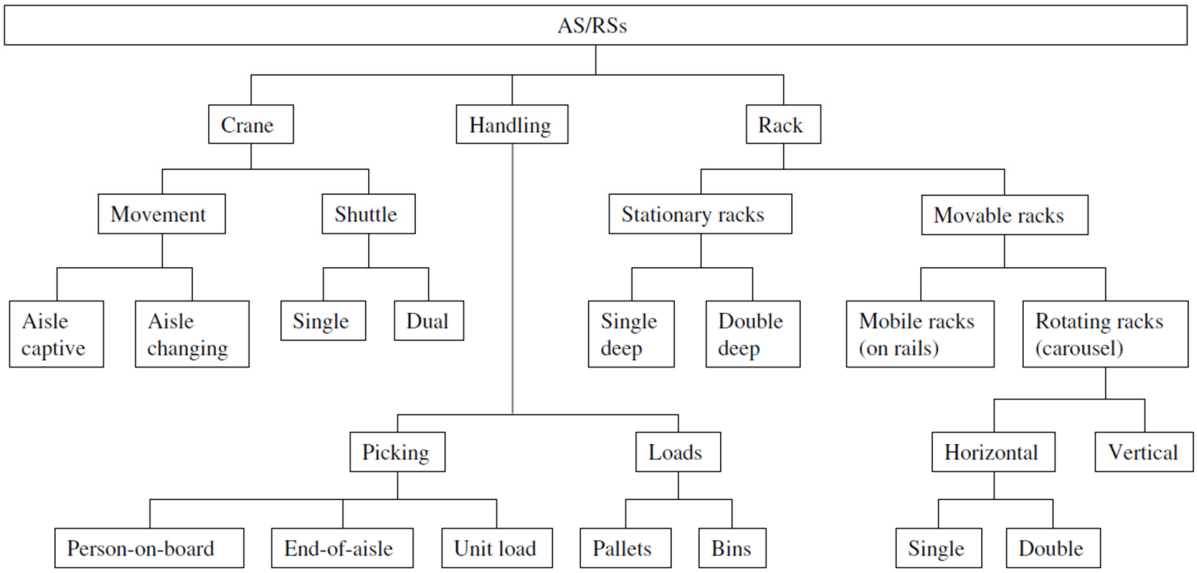

Figure 2.2 Classifications of various AS/RS system options (Roodbergen and Vis, 2009)

In order to meet the increasingly versatile demands posed by the rising of e-commerce, practitioners and researchers have taken one step further in redesigning the S/R mechanisms. Instead of using a single type of device (the crane) to perform all the movements and L/U processes, AS/RSs based on collaborative service provided by multiple S/R devices are developed. In such systems, S/R operations may be performed in multiple stages, possibly through collaboration between multiple types of S/R robotics. Transaction rates in such system are usually higher than traditional crane-based AS/RSs. Moreover, unlike the crane based AS/RSs in which the cranes are most likely unchangeable after the initial construction, S/R robotics here may be expandable and easily added or decreased according to current demands. Such flexibility benefits the industries with better cost-effectiveness when facing varying demands and unexpected changes. In addition, system robustness is improved in case that failure of a single device does not necessarily lead to the suspension of the entire system. Azadeh et al. (2019) reviewed the recent development of various types of robotized and automated warehouse systems and discussed critical issues involving both system design aspects and operational control aspects. Hamzaoui et al. (2021) studied three types of AS/RS: bi-directional flow-rack, multi-aisle, and mobile-rack, and developed efficient optimization method based on dominance properties for system design.

2.2 Autonomous Vehicle Storage and Retrieval System (AVS/RS)

Autonomous Vehicle Storage and Retrieval Systems (AVS/RS), the main topic of this research, represent one of the most preeminent innovations among all AS/RS variations. AVS/RSs are also called Multi-layer Shuttle Storage and Retrieval Systems. Ever since its first conceptualization by Malmborg (2002), AVS/RS has gained vast research interests from practitioners. An AVS/RS is characterized by horizontally operating vehicles sharing a fixed number of lifts for vertical movement (Malmborg, 2002). Compared with its traditional counterparts like crane-based automated storage and retrieval system (CBS/RS), AVS/RSs provide users more flexibility in system design and asset configuration by allowing the designer to change the number of vehicles operating in storage racks when demand changes. Just like all innovative efforts made since the introduction of AS/RS, variations to further improve system performance and reduce costs are studied. Malmborg (2003) further studied the interleaving dynamics of AVS/RS operations based on state equation model. Two configurations of AVS/RS are defined, namely *tier-captive* and *tier-to-tier* (Heragu et al., 2009, Marchet et al., 2013). Unlike the tier-captive configuration where each tier has one vehicle, in the tier-to-tier configuration vehicles ride lifts to move between tiers, and thus less number of vehicles are needed and costs are reduced. On the other hand, tier-captive configuration offers better throughput responsiveness performance as more vehicle resources are deployed. Hu et al. (2005) introduced and analyzed *split-platform storage and retrieval system* (SP-AS/RS), in which one vertical platform and N horizontal platforms serve N tiers of an AS/RS rack. Carlo and Vis (2012) introduced a variation to AVS/RS and named it *shuttle-based storage and retrieval system* (SBS/RS), which applies tier-captive vehicles in most cases and two non-passing lifts and buffer conveyors are used. Ning et al. (2016) studied a multi-elevator tier-captive SBS/RS design, in which the number of elevators (lifts) can be adjusted in system design.

2.2.1 Alternative Designs in AVS/RS Category

This research mainly focuses on AVS/RS designs in which each aisle has its own storage lift, retrieval lift, shuttles (vehicles), shuttle (vehicle) lift, and two I/O buffer conveyors are used on each tier. Thus, shuttles and all lifts are aisle-captive – the term Shuttle-based Storage and Retrieval System (SBS/RS) is used for such designs in the rest of this research. The storage lift and retrieval lift are installed at the same side of the aisle and interact with the encompassing

system (typically the conveyor system which delivers storage totes and takes away the retrieval tote). The shuttle lift, however, is installed on the other side, and only transfers vehicles from tier to tier when needed, and thus does not interact with other systems. However, it is necessary to mention that this is not the only design stereotype using separate vertical and horizontal S/R devices. Terminologies for similar devices or processes may also be different in those designs. Moreover, the S/R task patterns and thus cycle time models in those designs can be different.

AVS/RS with Cross-aisle Vehicles

In the first AVS/RS design introduced (Malmborg, 2002), vehicles are designed to travel along the x-axis on the aisles, and the z-axis on the cross-aisles (Figure 2.3, Figure 2.4), thus any vehicle is able to access any storage position in the system. Vehicles travel along end-of-aisle rails when transferring between aisles. For vertical movements, there is only one type of vehicle lift that carries a vehicle to/from the I/O at the floor level to pick up or drop off the product. Thus, a vehicle always travels with the lift in every storage/retrieval task – with the exception that S/R tasks at the floor level are processed by the vehicle alone. The number of vehicles is configurable, while normally no more than one vehicle is deployed to the same tier of all aisles in order to avoid collisions. The number of lift(s) is also a design decision depending on system throughput requirement. Due to this movement pattern, I/O buffers on each tier are not needed. All buffered loads and retrieval requests are "pooled" in a single queue. Such system is especially suitable for rack configurations where there is a large number of shallow storage aisles and provides good cost-effectiveness when the tasks are less time critical. However, the operational control of this system could be difficult due to its complexity caused by various interaction effects (e.g., vehicle-lift interactions, blocking effects within aisles, etc.) in the system.

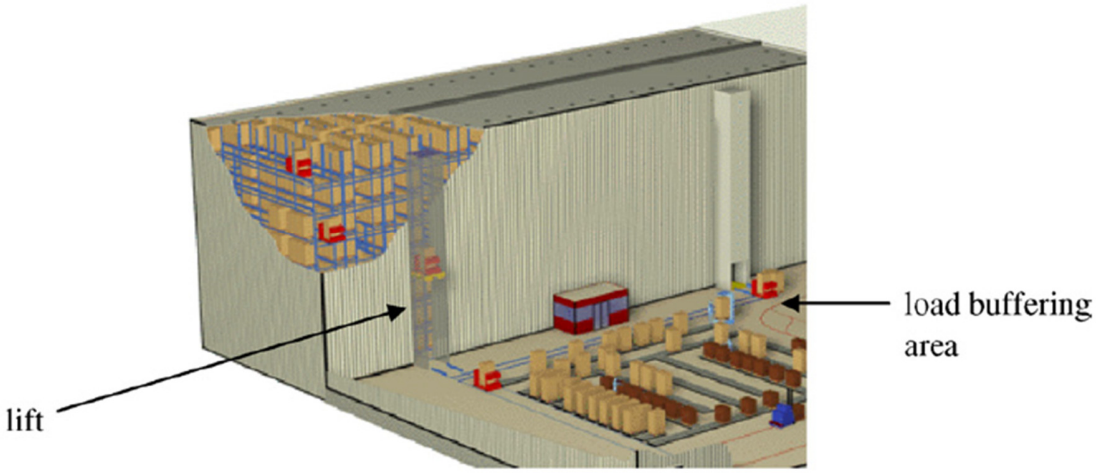

Figure 2.3 Illustration of AVS/RS (Malmborg, 2002)

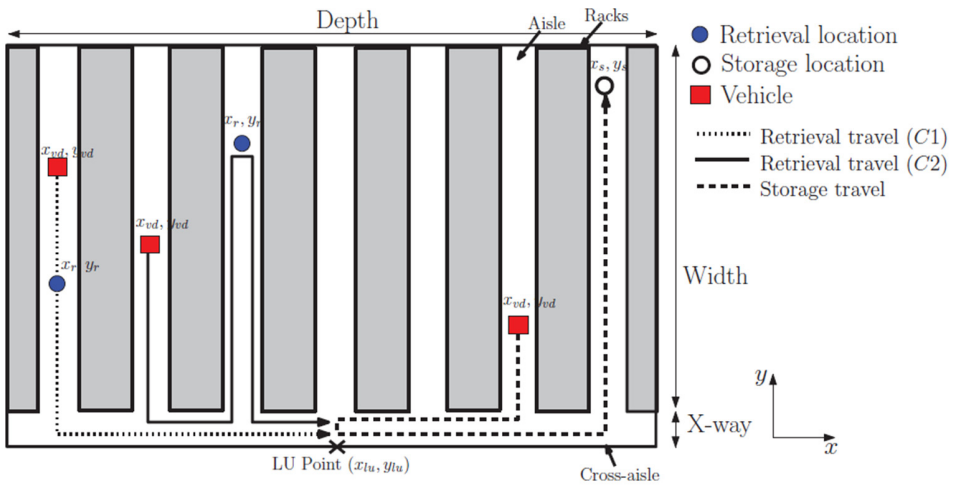

Figure 2.4 AVS/RS movement (Roy et al., 2012)

Shuttle-based Storage and Retrieval System (SBS/RS)

A SBS/RS design applies a number of shuttles (vehicles) and two non-passing lifts – a storage lift and a retrieval lift – for each aisle. Different from the original AVS/RS, in SBS/RS both the lifts and shuttles are aisle-captive, thus no cross-aisle operations are performed (Figure 2.5). Also, unlike in the original AVS/RS design, vehicles in SBS/RS are not carried by the S/R lifts from/to I/O tiers. Instead, separate I/O buffers are installed for lift-shuttle interactions on each tier of each aisle. In traditional SBS/RS each tier is deployed with a dedicated shuttle (tier-captive),

thus the number of shuttles is no longer configurable once the rack dimensions are determined. Such designs are effective in situations where the demand scenarios are less uncertain. Another type of SBS/RS applies tier-to-tier techniques, where shuttles are transferred between tiers when needed by an aisle-captive shuttle lift. SBS/RS is also named as Multi-shuttle S/R System by many researchers (Carlo and Vis, 2012) and manufacturers.

A Multi-elevator / multi-lift SBS/RS is a variation from typical SBS/RS (Figure 2.6). It is also tier-captive. Different from a typical SBS/RS in which lifts are installed on one side of the aisle, in multi-lift SBS/RS lifts are installed at different positions in the rack along the aisle direction. The multi-lift design balances the workload of shuttle carriers, while also introduces more design alternatives in terms of lift number and positions. This system is first studied by Ning et al. (2016).

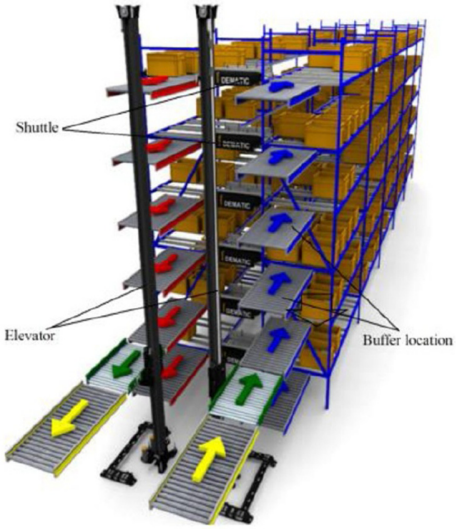

Figure 2.5 Front view of a single aisle of SBS/RS (Dematic Multishuttle 2 Whitepaper, 2013)

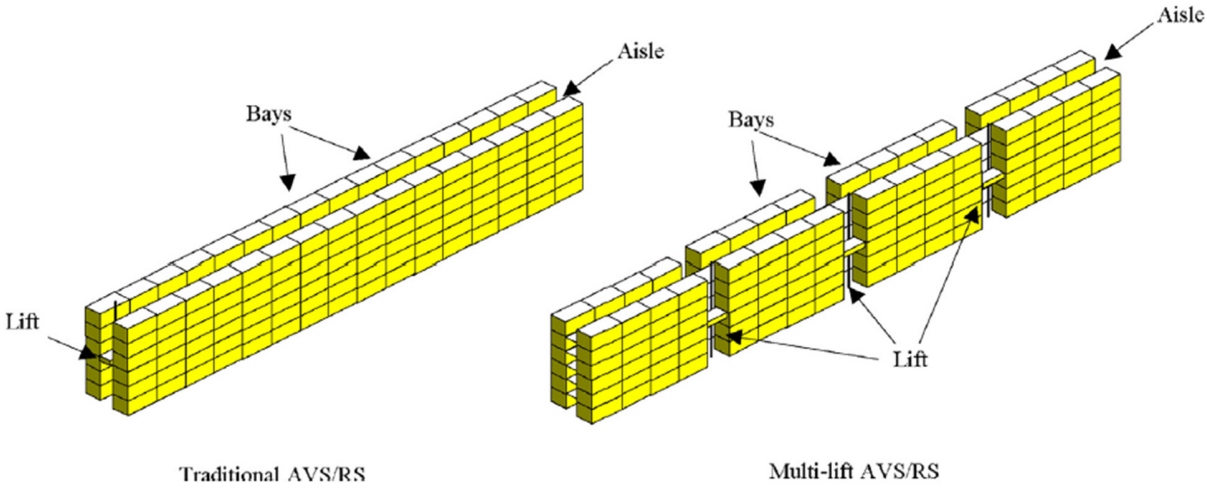

Figure 2.6 Difference between traditional SBS/RS and multi-elevator SBS/RS (Ning et al., 2016)

Split-platform AS/RS

In SP-AS/RS, one vertical platform and *N* horizontal platforms serve *N* tiers of an AS/RS rack (see Figure 2.7). This concept is similar to AVS/RS which applies separate lifts and vehicles to perform operations. The major difference is that, in an SP-AS/RS only the load is transferred from one platform to the other and allows for the other platform to perform the next stage of operation. The same tiers in two adjacent racks share a horizontal platform. This system is first introduced by Hu et al. (2005).

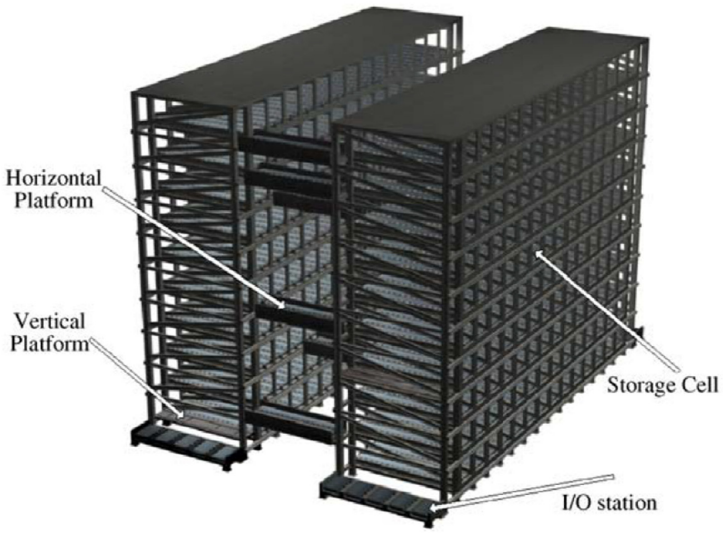

Figure 2.7 Split-platform AS/RS (Hu et al. 2005)

2.2.2 Design and Control of AVS/RS (and variations)

Design and control methods to optimize the performance of the AVS/RS are widely discussed, where analysis based on queuing theory are typically conducted. Most of those works focus more on the design/conceptualization aspect and made assumptions like FCFS in S/R device services, random storage, and simple order structures and stock keeping unit (SKU)-level characteristics to simplify the problem. Although AVS/RS studied by those researchers are not necessarily the same as ours (as discussed previously), their research methodologies and insights are certainly important and enlightening our study.

Just like in AS/RS researches, analytical modeling approaches, including queuing models, state equation models, and optimization models, etc., are applied by many researchers in AVS/RS configuration design. The advantage of analytical modeling is to quickly evaluate among numerous alternative system configurations and offer sufficiently accurate performance estimates, before looking into more detailed aspects like operational control strategies. Various researches have been conducted in developing more accurate, efficient, and comprehensive analytical models. In the first conceptualization research by Malmborg (2002), the portions of single cycle (SC) commands versus dual cycle (DC) commands are highlighted as critical inputs in system performance evaluation. As mentioned before, the AVS/RS studied there has one single lift responsible for transporting both the vehicles and the totes in each aisle. This is different from the system studied in our research, as three different types of lifts are used for storage, retrieval, and tier-to-tier operations, respectively. In the later research by Malmborg (2003), a state equation model is proposed to estimate the proportion of dual command S/R cycles using opportunistic interleaving in AVS/RSs. "Opportunistic interleaving" in this context means dynamically combining storage and retrieval transactions into dual cycles to reduce the average vehicle time per transaction. The model extended the conceptual tools developed from the author's previous work (Malmborg, 2002) which require an estimate of this proportion to predict system utilization and throughput capacity. Kuo et al. (2007) developed a computationally efficient cycle time model for AVS/RS, based on which vehicle utilization and system cost are estimated with good accuracy. Based on analysis of device movement elements, 12 service scenarios occurring under realistic operating assumptions are formulated in the model. Although the queuing approximations embedded in the model exhibit substantial errors, the model is sufficient for system

conceptualization studies and thus narrows the range of design profiles warranting more extensive simulation-based evaluation and validation. Kuo et al. (2008) further developed a cycle-time model based on queuing network approach for AVS/RS using class-based storage policies to guide system design conceptualization. The model is illustrated through various realistic case problems. Based on in-depth analysis of SC and DC movement elements in AVS/RS operations, Fukunari and Malmborg (2008) developed an efficient cycle time model for AVS/RS and compared its performance with crane-based AS/RS. An iterative estimation procedure is developed to estimate proportion of DC cycles. Simulation based validation shows that their model is adequately accurate for system conceptualization. Later, Fukunari and Malmborg (2009) estimated the system throughput and resource utilizations based on a network queuing approach, which overcame the computational disadvantages of the state equation model and the inflexibility of the nested queuing model and thus provide effective screening of candidate design profiles prior to more extensive simulation based validation. Zhang et al. (2009) proposed approximation strategies for transaction waiting times for AVS/RS with non-Poison arrival rates and non-exponential service times. Their procedure is illustrated through realistically sized problems and showed good analytical simplicity and computational efficiency. Hu et al. (2010) proposed load shuffling algorithms for split-platform AS/RS to relocate items to minimize response times. Heragu et al. (2011) developed effective travel time models for the vehicles and lifts as in AVS/RS and also for cranes as in traditional AS/RS, based on which system performance with different lift machines configurations are analyzed. Both systems are modeled as open queuing networks (OQN), which enables quick evaluation of alternative configurations of the two systems. Roy et al. (2012) modeled a Cross-aisle AVS/RS as a multi-class semi-open queuing network with class switching. Design variations including different cross-aisle zoning plans, number of L/U points, and vehicle assignment rules (selection priority to different types of transactions) are evaluated. To estimate performance measures of tier-captive SBS/RS, Marchet et al. (2012) presented an analytical model based on open queuing network approach. They pointed out that the transaction cycle time, specifically the waiting time, is the most critical component to be evaluated. They also highlighted the design complexity due to the combined effect of the kinematic behavior of vehicles and lifts, and the creation of queues deriving from the interaction of those two types of devices. Ekren et al. (2012) and Ekren et al. (2014) modeled AVS/RS as a semi-open queuing network model after defined all possible scenarios for storage and retrieval transactions and their probabilities, and applied a

matrix-geometric method for analyzing it. Lerher et al. (2015a) proposed an analytical travel time model for SBS/RS. Device operating characteristics like acceleration and maximum velocity are considered in their analytical model. Lerher (2016) studied travel time model for double-deep SBS/RS, in which two lanes of storage locations are on both side of the aisle. The rearranging of blocking totes (in case that a retrieval from the second lane is blocked by a tote stored on the first lane) is considered in the model. Ekren (2017) provided a graph-based solution, by which critical SBS/RS performance indicators under various design concepts are visualized and compared. Borovinšek et al. (2017) presented a multi-objective optimization model for SBS/RS design. Their model used Non-Dominated Sorting Genetic Algorithm II (NSGA II) genetic algorithm. Average throughput time, energy consumption, and total investment cost, are the three objective to be minimized. Ha and Chae (2019) developed a decision model to determine number of shuttles of the SBS/RS. Ekren (2020) presented a multi-objective optimization solution procedure to reduce cycle time and energy consumption of AVS/RS.

Modeling and Simulation techniques are applied by many researchers to support AS/RS design and control decision making. Ekren (2011) built simulation model for an AVS/RS using commercial simulation software Arena. Performance indicators including device utilizations, queue lengths, waiting times, etc., as well total cost, are measured. Simulation experiments are conducted to evaluate system performance under various configurations. Marchet et al. (2013) investigated main design trade-offs for a tier-captive AVS/RS using simulation, and developed a comprehensive design framework. Their simulation results suggested fewer and longer aisles in order to reduce investment costs, and suggested to target at device utilization level of 90% so that waiting time is acceptable. Lerher et al. (2015b) presented simulation analysis for SBS/RS and explored the effects of various design assumptions. Tappia et al. (2017) developed queuing models which can handle both specialized and generic shuttles and both continuous and discrete lifts of multitier shuttle-based compact storage systems, and validated their models using simulation. Ning et al. (2016) developed simulation model for a multi-elevator SBS/RS and conducted a specific case study to evaluate alternative designs with different number of elevators. Akpunar et al. (2017) applied simulation techniques to explore energy minimum AVS/RS warehouse design providing maximum utilization of resources in the system. Ekren (2020) developed a simulation-based experimental design for SBS/RS warehouse design by considering energy related performance metrics.

2.3 Coordinated Control of Warehouse Order Fulfillment System

As picker-to-part systems, automated warehousing systems are closely associated with downstream order picking processes through which pickup orders are finalized. The bins (totes) containing SKUs requested by an order are first retrieved from the storage system, then unloaded to a conveyor system and delivered to picking operators. The bin to be picked then stops in front of the picking operator and is processed by the operator according to order information (typically updated to the operator by the warehouse management system). After the pickup, the tote may be returned to storage by the conveyor system if it is not empty or may leave the system otherwise. The replenishment of SKUs (and possibly order containers) is another important process to be fulfilled by the conveyor system and the storage system.

2.3.1 System Structure and Design

As summarized by Boysen et al. (2019), an order fulfillment system based on picking workstations consists three types of element systems:

- ***Storage system***: the AS/RS where bins containing SKUs are stored;
- ***Conveyor System***: the intermediate system which delivers requested bins between storage system(s) and station(s);
- ***Picking workstation***: in which requested SKUs are withdrawn by a human picker from delivered storage bins to fulfill customer orders. The withdrawn SKUs are normally placed into empty containers (order bins), according to SKU types and quantities requested by customer orders.

Depending on the application environments, different warehousing systems face different order structures characterized by order size, SKU variety, and time constraints, etc. Multiple picking workstations and complex conveyor design are common in today's automated warehousing systems. Thus, the storage system must be wisely configured considering the functionalities and performances of the material handling systems encompassing it in conceptual design phase, and also coordinated with inbound and outbound flows from/to those encompassing systems during operational control. The true performance of an AS/RS is typically influenced by the other material handling systems in the warehouse as are the other systems' performances influenced by the AS/RS (Roodbergen and Vis, 2009). In many applications in manufacturing

environment, the AS/RSs are responsible to transfer and sequence production materials between machines. In typical e-commerce applications, the AS/RSs need to coordinate with complex transfer, pick-up, sorting and packaging processes to fulfill various customer needs. Thus, the design and control of AS/RSs need to be addressed together with the design and control of those encompassing systems. Amato et al. (2005) suggested a control architecture for management of automated warehouses. De Koster et al. (2007) investigated different methodologies of design and control of warehouse order picking, multiple aspects the systems are discussed for both picker-to-order and order-to-picker systems.

2.3.2 Design and Control of Order Picking Workstations

Order picking workstations are able to handle a wide spectrum of SKUs and provide high service rate up to 1000 order lines per hour (De Koster et al., 2007). Studies have been conducted to improve workstation performance from design and control aspects. Andriansyah et al. (2010) proposed a simulation modeling approach to study order picking workstation performance in automated warehouses. The order picking workstation is viewed as a polling system which interacts with the storage system through multiple buffer conveyors and a return conveyor. The authors then developed an aggregate model which provides satisfactory accuracy predicting both tote and order flow times. Based on the same system configuration, Claeys et al. (2016) conducted queuing analysis to compute stochastic bounds for order flow times. Andriansyah et al. (2014) studied design and control of a workstation which retrieves SKUs from an AS/RS in a larger order-picking system. The workstation receives multiple orders simultaneously, and four picking policies are designed and evaluated using simulation techniques. They also proposed a novel design of carousal-type conveyor system to avoid deadlocks between different types of bins.

2.3.3 Synchronization of Warehousing Workflows to Improve Overall Performance

Instead of focusing on the storage systems in isolation, improving the overall performance of the warehouse system has attracted increasing number of researchers. Most of those researches assume the storage system is either crane-based AS/RS or not explicitly specified. Eben-Chaime (1996) proposed an integrative simulation model in which warehouse activities interact with other functions of the total system. In the author's model, the storage requests are viewed as dependent to prior retrieval requests to present the pickup-return process. A hybrid command mode is proposed which outperformed dual-cycle mode under this condition. Chincholkar and Chetty

(1996) applied simultaneous job scheduling to an AS/RS system and machines in a flexible manufacturing system using Petri Nets and the Taguchi method. Van Gils et al. (2018) reviewed and classified order picking planning problems – both tactical ones (zoning, storage assignment, etc.) and operational ones (batching, job assignment, etc.) – in picker-to-part systems. They pointed out that optimizing those problems sequentially may yield a suboptimal overall warehouse performance due to their interactions, and thus suggested development of good policy combinations. In a study by Füßler and Boysen (2019), the authors aimed at concerted processing of picking orders and storage bins delivered from the crane-based AS/RS to picking workstations. They pointed out that a careful synchronization of delivered and demand SKUs can reduce the workload of the AS/RS, and formulated an optimization problem and develop a heuristic solution procedure.

On the other hand, research specifically focusing on synchronization/coordination of AVS/RS-based warehousing systems also continues. Tappia et al. (2019) investigated the interactions between downstream picking systems and alternative upstream storage systems – crane-based AS/RS versus SBS/RS (tier-captive). They developed generic semi-open queuing models for both alternative designs, and proved that the SBS/RS yields more savings in investment costs comparing to the traditional AS/RS as less storage aisles and picking stations are needed, paired a lower total throughput time at a given order arrival rate. Li et al. (2019) built simulation model for a warehouse system consists of subsystems including multi-zone AVS/RS, pickup workstations, and conveyor network. They studied the interactions between those subsystems and made coordinated control attempts to improve overall system efficiency and reduce blocking.

2.4 Summary

Although the SBS/RS in our study is quite different from most of the AS/RSs introduced in this chapter in terms of design and operation, the abundant research methodologies and approaches proposed in the AS/RS-related literature provide us important insights and valuable experience. We attempt to identify critical factors which are in common between our system and AS/RS in general, and discuss about the insights we learned. More specifically, they are:

1) **Requirement identification and system configuration in conceptual design.** The major requirement is typically the storage capacity determined based on demand

evaluation, and (potentially) constrained by facility layout. System configuration parameters including number of aisles/tiers/columns, etc. are decided and usually unchangeable in the future. Analytical methods developing travel-time models are mainstream approaches which rapidly evaluate thousands of alternative system configurations and select the one which is expected to best fulfill the design objectives (throughput, cycle time, and cost, etc.). However, as system complexity and demand uncertainty increase, the accuracy of analytical models is reduced. Simulation modeling enables further analysis of the candidate configurations with better accuracy and traceability. As a simulation model is typically costly to build and often lacks generality, simulation approaches should be better conducted after a smaller search space is obtained through "rough-cut" evaluations by analytical approaches.

2) **Use of control strategies to improve operational performance**. As the demand is typically stochastic and usually dynamic, control approaches need to be implemented to continuously ensure system's operational performance and robustness. Those control strategies involves multiple aspects of system control, and typical approaches are top-down optimization algorithms (e.g. wave/block sequencing of requests), bottom-up dynamic policies (e.g. request prioritization rules), as well as static rules (e.g. class-based storage, dwell policies of S/R device). Simulation approaches have become more and more significant in evaluating the overall performance of a combination of various control aspects. On the other hand, analytical approaches are still important here to provide mathematical validity of the control approaches, and offer insights to system design phase.
3) **Synchronization with encompassing subsystems**. Researchers have long recognized AS/RS's design and control complexity due to interactions with encompassing subsystems, and approaching to improve the overall performance of the larger warehousing system. Design and control of pick workstations and conveyor systems in AS/RS-based systems are studied. Researches focusing on control coordination between all the subsystems, as well as the effects of different AS/RS application environments (e.g. distribution center vs. production center) are still very few.

As a growing research area, most AVS/RS studies till now focused on either introduction of new designs or/and approaches for design configuration. The various analytical approaches and

simulation studies proposed by those researchers highlighted the unique operational characteristics of AVS/RS and provided excellent insights, which significantly enlightened our study. On the other hand, although many AVS/RS variations are introduced and studied by those forerunning researchers, we did not find much researches specifically focusing on the tier-to-tier SBS/RS as introduced in our study. The majority of related researches are about the AVS/RS with single-type lift(s) and cross-aisle vehicles. For the SBS/RS section, researches till now mostly focused on tier-captive designs. Furthermore, researches on operational control of AVS/RS are still quite few – not to mention the SBS/RS subset and the synchronization of the larger warehousing system based on AVS/RS. We view all those as our research opportunities to contribute the body of knowledge by developing a comprehensive methodology to guide the design, control, and coordination of SBS/RS warehousing systems.

Chapter 3 SBS/RS Simulation Model Design and Development

3.1 Overview

A data-driven and data-generated simulation model is built to support development of both the design and configuration methodology and operational control strategy of SBS/RS-based warehouse systems. The operations in the SBS/RS including device movements, acceleration and loads/unloads, as well as the dynamics in task, tote and SKU (stock keeping unit) information, etc., are simulated precisely using a hybrid simulation methodology based on Discrete Event Simulation (DES) and Agent-Based Simulation (ABS) techniques. The simulation models are implemented in AnyLogic (Grigoryev 2015), which is a commercial multimethod simulation modeling tool. The model simulates the systems' demand scenarios either described by stochastic task arrival process based on user-configurable order structure and SKU-level characteristics parameters, or described by predefined task sets and inventory information. For the design and configuration aspect, the simulation model is auto-generated with predefined input data of design parameters including the numbers of aisles, tiers, columns, depths, shuttles deployed, etc., as well as physical parameters including rack dimensions, device velocities and accelerations, load/unload time, etc. With the custom system objects and modeling mechanisms we developed using AnyLogic, the simulation model is generated automatically given the input data. For the operational control aspect, control decisions involving storage assignment as well as task scheduling for different devices, are identified and implemented in the simulation model. Moreover, various user-configurable control algorithms are developed for different control decisions. Finally, the simulation model provides a solid experiment platform to simulate combinations of demand scenarios, designs and configurations, and control strategies, to evaluate system performance by critical indicators presented in numerical and graphical forms.

3.2 Simulation Modeling Approach

To develop the simulation model, an iterative three-step methodology that includes Domain Modeling, Conceptual Modeling and Simulation Modeling is applied. Domain and Conceptual Models are important prerequisites for the simulation modeling approaches. The Domain model describes the problem domain and the conceptual model further describes the domain in language-independent simulation terms. As illustrated in Figure 3.1, system characteristics are identified and developed into simulation forms gradually in each step of the methodology, and reworks of the previous steps are performed whenever inconsistencies and discrepancies are observed. This iterative modeling methodology ensures the simulation model is a valid and acceptable abstraction of the practical SBS/RS-based warehouse systems to be analyzed and controlled under typical design and operational environments. As the system characteristics identified through the Domain and Conceptual Modeling steps are consistent with the system introduction and problem statements in the previous chapters, we will not discuss those steps in detail here.

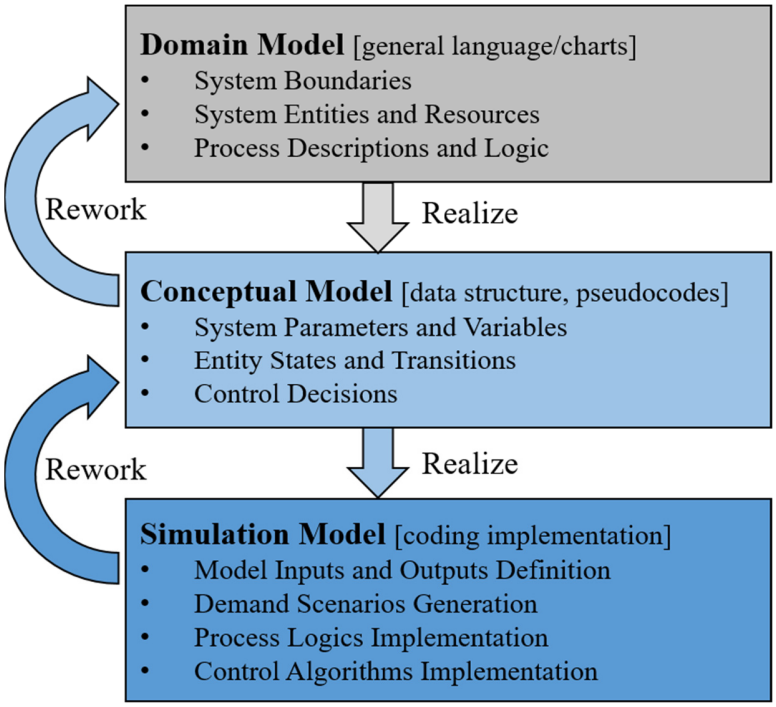

Figure 3.1 The Iterative Domain-Conceptual-Simulation Modeling Methodology

In the simulation models, 3D animations and various output statistics are developed and used to facilitate the analysis of the systems' dynamics as well as the verification of the control approaches.

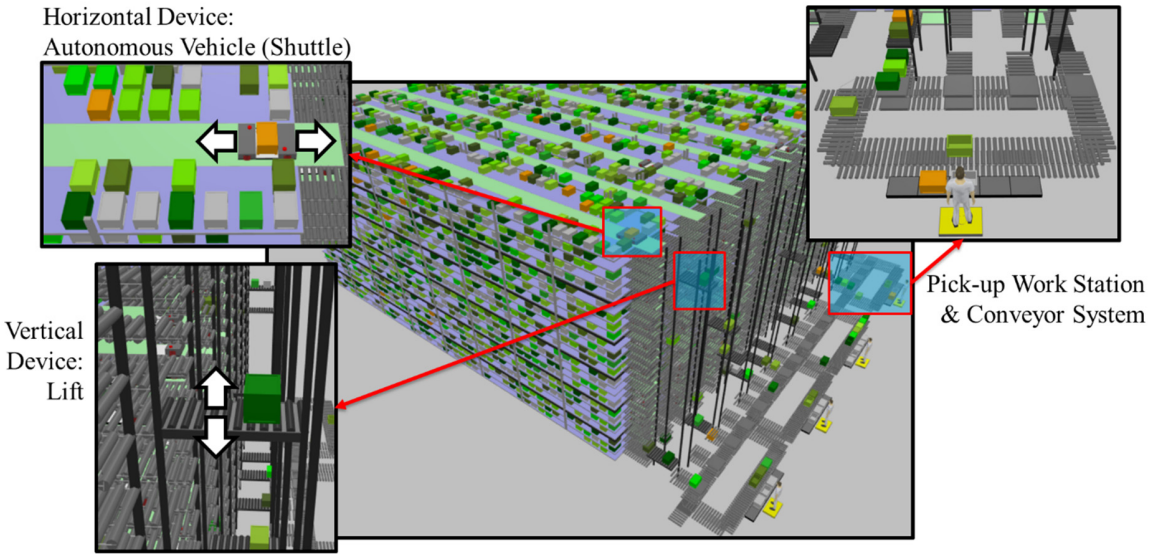

Figure 3.2 SBS/RS-based warehouse system simulation model (built with AnyLogic)

3.2.1 Modeling System Objects and Processes

The simulation model is auto generated with predefined input data of design parameters including the numbers of aisles, tiers, columns, depths, shuttles deployed, etc., as well as physical parameters including rack dimensions, device velocities and accelerations, load/unload time, etc. Discrete Event Simulation (DES) techniques are selected as the main methodology for developing the simulation model, while some Agent-Based Simulation (ABS) techniques are also applied for modeling the low-level device-task interactions. Each *aisle* of the SBS/RS is modeled as an individual service agent that contains a set of S/R devices, a rack storage area which is a set of *slots* indicated by (x, y, z) coordinates, and the roller-conveyor *input/output buffers* on different tiers. The aisle's S/R devices including *shuttles*, *storage lift*, *retrieval lift* and *shuttle lift* are modeled as agents each has specified logics for providing storage and retrieval services, while the device service times are simulated under the DES framework. Generally speaking, the service times of all devices can be described in two aspects: the times needed for load/unload operations, and the rectilinear travel times. Given a service process described as a set of movement, load and unload commands, device service times are simulated deterministically under the DES framework:

the devices' L/U times are modeled as constant parameters, and the rectilinear travel time between every two locations i and j (either horizontal or vertical) with distance l are computed precisely given the device's velocity v, acceleration a and deceleration d as follows:

$$\tau_{i,j} = \begin{cases} \sqrt{2l\left(\dfrac{1}{a+\dfrac{a^2}{d}}+\dfrac{1}{d+\dfrac{d^2}{a}}\right)}, & if\ l < 0.5v^2\left(\dfrac{1}{a}+\dfrac{1}{d}\right) \\ \dfrac{v}{a}+\dfrac{l-0.5v^2\left(\dfrac{1}{a}+\dfrac{1}{d}\right)}{v}+\dfrac{v}{d}, & Otherwise \end{cases}$$

The device travel times of all feasible paths (either horizontal or vertical) are pre-computed based on the input data during model generation.

Totes are defined as the basic entities in the simulation model, either stored in slots or transported by the S/R devices. The initial tote inventories in the aisles' rack storage areas are either auto-generated according to rack utilization parameter settings, or predefined in the input data. *Tasks* represents the minimum demand entities describing the information required for the S/R devices to provide services and the corresponding totes. A task can either be a *storage* task or a *retrieval* task, both require services from more than one S/R device. Moreover, under control assumptions where SKU-level characteristics are considered, SKU information is modeled as parameters/variables related to totes and tasks. Table 3.1 summarizes the major system objects and their processes identified for simulation modeling. Details of the service processes and control strategy options will be introduced in Section 3.2.3 and further discussed in Chapter 4 and Chapter 5.

Table 3.1 Major Objects in Simulation Modeling

Object	**Primary Parameters**	**Primary Variables**	**Primary Processes / Interactions**
Tote		SKU type & qty. Current [Task]	The basic entity of all process flows in simulation
Aisle	Rack size: X, Y, Z; Shuttles deployed: J	Rack utilization; SKUs inventory	$\begin{cases} J = X, & Tier-captive \\ J < X, & Tier-to-tier \end{cases}$
Slot	Location$[x, y, z]$	State ∈ {Idle; Occupied[Tote]; Reserved[Task]; etc.}	In 2-deep aisles ($Z = 4$), a slot in deep positions ($z = 3\ or\ 4$) cannot be accessed if its neighboring slot ($z = 1\ or\ 2$) is occupied

Input Buffer (IB)	Tier[x], Capacity	Totes {[Tote]}	Blocks future storage lift processes if IB is full
Output Buffer (OB)	Tier[x], Capacity	Totes {[Tote]}	Blocks future shuttle retrieval processes if OB is full
Task	Type ∈ {S, R}; Tote [Tote]; Target [Slot]; Arrival time; Due date; Precedence[Task]	Waiting Time; Cycle Time; State ∈ {V, SL, RL, IB, OB, etc.}	Tasks are realization of system demands, arrive to the aisles either in stochastic patterns or as defined by input data
Shuttle (V)	Velocity & acceleration; L/U times	Location [x, y]; Current [Task]	Storage (S): *[To IB→Load→To target→Unload]* Retrieval (R): *[To target→Load→To OB→Unload]* 2-deep Relocation (Re): [To blocker→Load→To target→Unload]
Storage Lift (SL)	Velocity & acceleration; L/U times; Tour capacity	Location [x]; Current {[Task]}	Storage (S): *[To I/O→Load→{To target→Unload}]*, "{}" indicates multiple tasks in the same tour
Retrieval Lift (RL)	Velocity & acceleration; L/U times; Tour capacity	Location [x]; Current {[Task]}	Retrieval (R): *[{To target→Load}→To I/O→Unload]*, "{}" indicates multiple tasks in the same tour
Shuttle Lift (VL)	Velocity & acceleration; L/U times	Location [x]; Current [Task] Current [Shuttle]	Tier-transfer (TT): *[To shuttle→Load→To target→Unload]*

3.2.2 Modeling Demand Scenarios

Demand scenarios are defined as the task arrival patterns to the aisles, either described by stochastic task arrival process based on a user-configurable order structure and SKU-level characteristics/parameters, or described by task sets and inventory information predefined in the input data. In the former case, the arrival processes of storage tasks and retrieval tasks to each aisle are defined in terms of inter-arrival time (IAT) distributions. The IAT distributions are defined as general random, non-negative distributions, each described by distribution mean (μ) and standard deviation (σ). When the focus is analyzing the systems' steady-state performances, the arrival rates of storage tasks and retrieval tasks must be equal, thus there is $\mu_S = \mu_R = 1/\lambda$ in which λ is the average task arrival rate for both task types ($\lambda = \lambda_S = \lambda_R$). On the other hand, there is not necessarily $\sigma_S = \sigma_R$ because σ_S and σ_R are measuring the variance of two different processes at the upstream of the system: σ_S is corresponding to the patterns that the storage totes are delivered

to the aisles (e.g. from conveyor networks or/and docking areas), while σ_R is corresponding to the patterns that retrieval requests are released to the aisles (e.g. by higher-level systems like MRP). In this research, we use lognormal distributions to approximate the general distributions of the IATs, because the lognormal distribution can be described by the mean and standard deviation to flexibly describe different variances of storage and retrieval IATs, and only gives positive values. In the simulation model, each of the two general distributions are approximated as an individual lognormal distribution $Lognormal(\mu_N, \sigma_N)$, where μ_N, σ_N are the mean and standard deviation of the included normal distribution, computed given the objective μ, σ as follows:

$$\mu_N = 2\ln(\mu) - 0.5\ln(\sigma^2 + \mu^2)$$

$$\sigma_N = \sqrt{-2\ln(\mu) + \ln(\sigma^2 + \mu^2)}$$

As introduced in Chapter 1, the performances of the SBS/RS are measured by the throughput indicator (maximum sustainable task arrival rates of both storage and retrieval tasks) and the responsiveness indicator (task cycle times and tardiness, etc.). With some specific control strategies, the performance of a same SBS/RS design and configuration could vary significantly under different demand scenarios. For example, Closest-Open-Location policy (COL) is a classic storage assignment approach studied in this research which always assign storage totes to closest available slots to the I/O buffers – when COL is applied, the task service times are largely affected by rack utilization (number of occupied slots divided by total slots, denoted as ρ). Obviously, the dynamics of rack utilization of each aisle is subject to the arrival patterns of storage and retrieval tasks. Modeling these patterns as individual IAT distributions does not present the typical operational environment of the SBS/RSs, because in practical operations, correlations exist between the arrivals of storage tasks and retrieval tasks. However, such correlations are very difficult to identify precisely because they are usually originated both from the nature of the SBS/RSs' application environments (e.g., those correlation could be very different between E-commerce applications and manufacturing applications), and from the configurations and performances of the SBS/RSs' upstream systems (e.g., the conveyor networks, MRP).

Because this research is targeted at developing universal design and control methodologies for SBS/RS-based warehouse systems, we are not trying to explore all the possible situations in

various customized application environments. Instead, three alternative solutions are proposed and implemented in the simulation model to address such complexity:

1) No explicit modeling of SKUs, while approximating the storage-retrieval correlation by controlling the numbers of storages and retrievals within specific time windows;
2) Model SKUs explicitly, and model the demand and replenishment patterns as random processes based on SKU types;
3) For manufacturing environment applications. Based on 2, model manufacturing orders as sets of retrieval tasks with precedence constraints. Model the additional pickup and return-to-stock patterns upstream and downstream of the SBS/RS as random processes for each order.

In this research, we primarily focus on the first solution. The storage-retrieval correlation complexity is addressed by assuming an expected rack utilization $\overline{\rho}$, as well as a time window T so that the total number of storage arrivals equals the total number of retrieval arrivals within each period of length T. The SKU-level characteristics of tasks and totes are not modelled explicitly, while to some extent T can be interpreted as the expected replenishment lead time of overall SKUs (in term of totes). The initial rack utilization is set to $\overline{\rho}$. The task arrival times of each duration are predetermined at the start of the duration as follows:

WHEN $(time = start\ of\ duration)$

$N = T\lambda$ *# Number of arrivals to be generated within period for each task type*

$Generate\ N\ storage\ arrival\ times\ \boldsymbol{t^S} = \left[t_1^S, t_2^S \dots t_N^S\right]\ from\ Lognormal(\lambda, \sigma_S)$

$Generate\ N\ retrieval\ arrival\ times\ \boldsymbol{t^R} = \left[t_1^R, t_2^R \dots t_N^R\right]\ from\ Lognormal(\lambda, \sigma_R)$

$Scale\ arrival\ times: \begin{cases} t_i^S = t_i^S \times T / t_N^S \\ t_i^R = t_i^R \times T / t_N^R \end{cases}, \forall i \in [1 \dots N]$

$Continue\ Simulation\ based\ on\ \boldsymbol{t^S}, \boldsymbol{t^R}$

Using this approach, the rack utilization equals $\overline{\rho}$ at the end of each time window, and the resulting IAT distributions of storages and retrievals are approximately the same with their original lognormal distributions. Because the rack capacity of an SBS/RS aisle is typically large, based on multiple simulation experiments we found it reasonable to view the resulting time-average of $\rho \approx \overline{\rho}$ with this approach. It is noticeable that both $\overline{\rho}$ and T are simulation inputs specified by the

decision maker to present the "typical" demand scenario(s) and/or "routine" warehousing operations that the decision maker interested in. Generally speaking, the variation of ρ is larger when the rack capacity XYZ is smaller, and also larger when either the IAT standard deviations σ_S and σ_R or the time window T gets larger.

3.2.3 Simulating Operational Control Decisions

Operational control decisions involving storage assignment as well as task scheduling for different devices, are identified in Table 3.2 and implemented in the simulation model as illustrated in Figure 3.3. Generally speaking, all these control decisions can be interpreted as "select one from multiple candidates". Those control decisions can be made either based on bottom-up approaches like dynamic dispatching rules, or through top-down approaches like mathematical programming – or even some heuristics that combine the advantages of both. The development and evaluation of the control strategies will be discussed in depth in Chapter 5. Moreover, for each control decision, various alternative and user-configurable control approaches are identified and implemented in the simulation model, as listed in Table 3.2.

Table 3.2 Operational control decisions in an SBS/RS aisle

Control Decision	**Selection Candidates**	**Alternative Approach Examples**

D1:
Storage Assignment
	Available **{slots}** in the aisle
	[select one slot for each storage task arrived to the aisle]
		• Pure Random Storage (PRS)
		• Closest Open Location (COL)
		  ■ Consider horizontal distances only
		  ■ Consider both horizontal and vertical distances
		  ■ Prioritize 2-deep slots
		• By number of available slots on tiers (largest)
		• By input buffer sizes of target tiers (smallest)
		• By SKU characteristics
		  ■ Predetermined zoning by SKU turn-over rates
		  ■ By correlations of SKU-types

D2: Retrieval Lift Scheduling	Retrieval **{tasks}** that completed shuttle services and unloaded to the output buffers [select one task as the next task for retrieval lift service]	• First Come First Serve (FCFS) • Closest to R.Lift's current location • Earliest Due Date (EDD) • By output buffer sizes of target tiers (largest)

D3: Shuttle Scheduling	Storage **{tasks}** that completed storage lift services and unloaded to the input buffers; Retrieval **{tasks}** arrived to the aisle. [select one task (either type) as the next task for shuttle service]	• First Come First Serve (FCFS) • Closest to shuttle's current location • Earliest Due Date (EDD) • Dual Cycle Interleaving (DC) • Makespan Minimization Heuristics ■ A* Search Algorithm
D4: Relocation (for 2-deep racks)	Available **{slots}** on the tier of each retrieval task arrived but blocked by a tote (blocker) in the neighboring slot. [select one slot for each blocker]	(Similar with D4 except that the search scope is within one tier)
D5: Tier-transfer (for tier-to-tier systems)	1. **{tiers}** that have shuttle tasks but without shuttles 2. **{shuttles}** that are idle [select one tier and one shuttle for each transfer-service]	• By number of tasks on tiers (largest) • By expected shuttle travel times (smallest) • Resource Allocation Heuristics ■ $P\|\|C_{max}$ Algorithm

3.3 Model Verification and Validation

Verification and validation (V&V) of the simulation model are performed using the aforementioned iterative Domain-Conceptual-Simulation modeling methodology. In simulation modeling, *verification* refers to the process of determining that a model implementation and its associated data accurately represent the developer's conceptual description and specifications, and *validation* refers to the process of determining the degree to which a simulation model and its associated data are an accurate representation of the real world from the perspective of the intended uses of the model. The verification of the simulation model is primarily based on various system performance statistics (either real-time or tally) collected by the model (Figure 3.4). Other model verification approaches, including code walk-throughs, visual verification using the model animations, and cross-verification with the queuing-based analytical model developed in Chapter 4, where also performed continuously throughout the research.

In addition, during the 18-month research project with our industry partner, the simulation model is continuously validated by our sponsors who manufacture SBS/RS products and provide automated warehousing solutions to their customers in various industries. The validation approaches in this research are primarily can be described in three aspects:

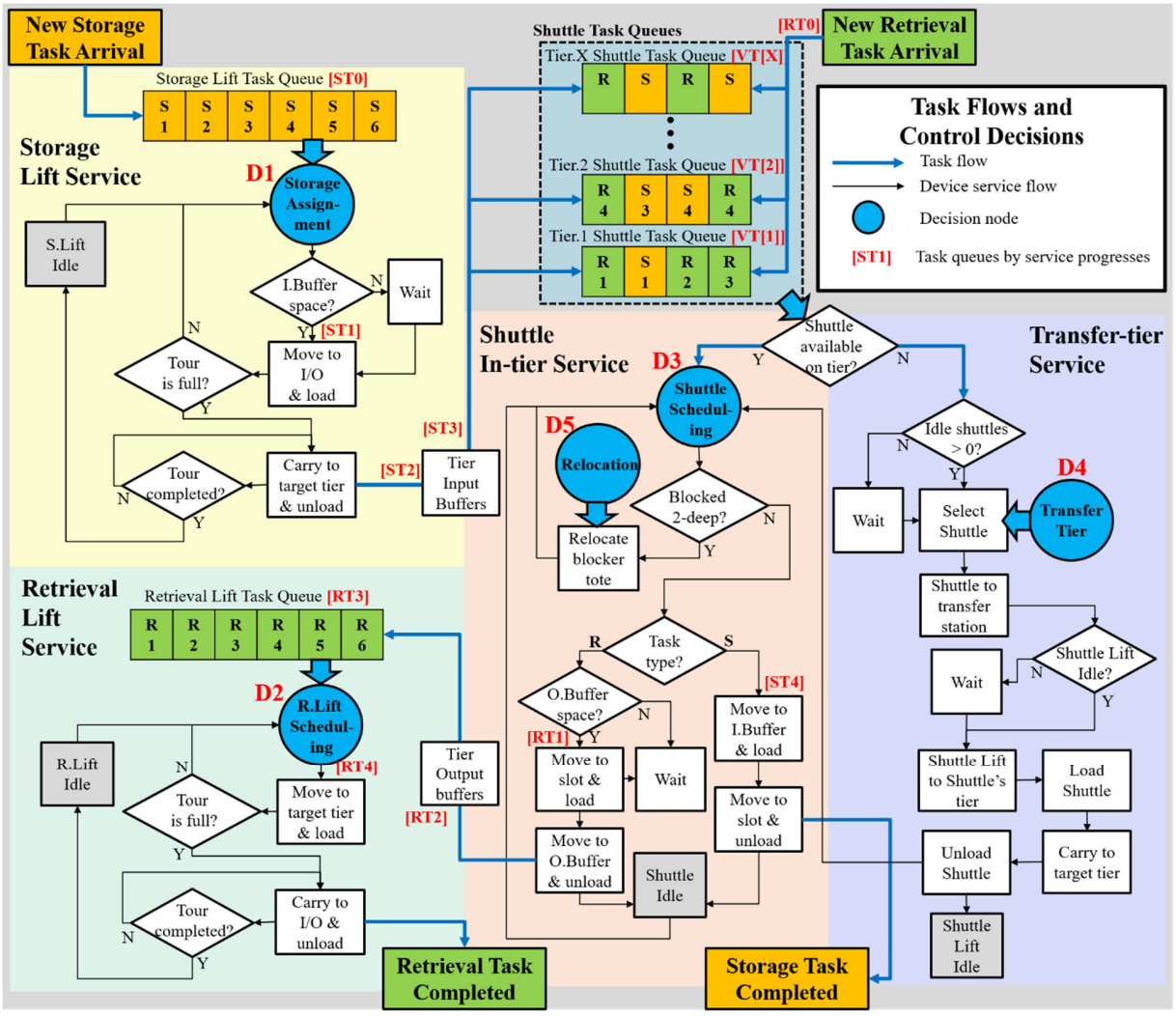

Figure 3.3 Implementation of the operational control decision in the simulation model

1) **System parameters**. The physical parameters of the racks (both 1-deep and 2-deep) are valid presentations of the sponsors' standard design criteria. Devices' velocity, acceleration/deceleration and L/U time parameters are determined based on the actual device performances tested by the sponsors. Miscellaneous design factors, for examples the existence of maintenance floors (which makes the vertical distances between tiers non-uniform), and devices' response delays from the Warehouse Control System, and buffer conveyor capacities and speeds, are also considered in the simulation model according to the needs of the sponsors.
2) **Demand formulation**. Although practical demand information from the sponsors' customers is not accessible due to confidentiality reasons, our formulation of S/R tasks and IAT

distributions in Section 3.2.2 are validated by the sponsors as acceptable approximations of the practical operational environments. The analytical model developed in the conceptual design approach further cross-validated the simulation model – to be discussed in Chapter 4. Moreover, the simulation model can also take deterministic tasks (deterministic arrival times, and either predetermined storage assignment or not) described in spreadsheet format as model inputs to facilitate V&V.

3) **Device control**. The general service processes and the control decisions are validated by the sponsors. The control algorithms are verified and validated through systematic development approaches based on both top-down approaches and bottom-up approaches – to be introduced in Chapter 5. Besides the five control decisions for scheduling and storage assignment identified in Section 3.2.3, other control rules (e.g., device dwelling policies) are also implemented as options in simulation although they are not the primary focus of this research.

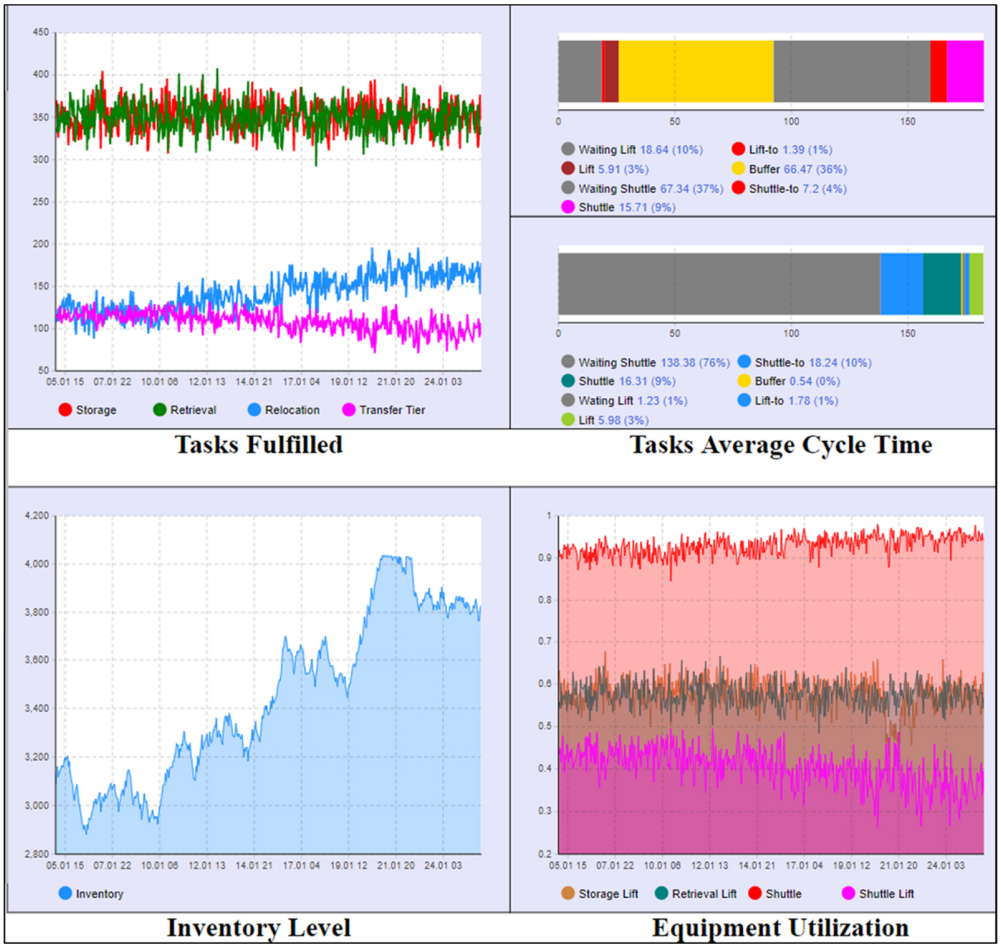

Figure 3.4 System performance statistics in the simulation model

It is noticeable that the model validation is an on-going task that accommodates the future requirements proposed by the sponsors and/or decision makers. For example, industrial users can continue this process incorporating their own proprietary/confidential data and design/configuration criteria, and explore design and control methodologies for their own business in further research.

3.4 Using the Simulation Model

According to a survey conducted by Smith (2003), the application of simulation technology in manufacturing systems can be classified into three classes: 1) for system design that involves long-term decisions and the analysis of design alternatives; 2) for system operations that involve short-term decisions including operations planning and scheduling, real-time control, operating policies, and performance analysis; and 3) simulation language/software package development. The simulation model in this research is primarily developed to support the first two classes: the system design decisions (conceptual design) and the system operation decisions (control strategy development) of SBS/RS warehouses, as illustrated in Figure 3.5. These two topics will be explored in detail in Chapter 4 and Chapter 5, respectively. On the other hand, although the simulation model is developed on a specific commercial simulation software (AnyLogic), the modeling approaches applied here are generalized and most of the model logic components are coded in Java language. Thus, the simulation techniques here are considered as compatible with different simulation platforms, feasible for software packaging, and potentially expandable to similar warehousing systems of larger scopes or/and applying different AS/R technologies.

3.5 Summary

The generic, data-generated simulation model was developed according to an iterative Domain Modeling, Conceptual Modeling and Simulation Modeling methodology. System objects and service processes, demand scenarios, as well as control decisions are modeled, verified and validated. Demand scenarios are modeled based on random inter-arrival-time distributions of storage and retrieval tasks, and alternative modeling solutions are proposed to handle the storage-retrieval correlations and SKU-level characteristics. Five types of control decisions – storage assignment, retrieval lift scheduling, shuttles in-tier scheduling, relocation, and tier-transfer – are identified. Multiple alternative control approaches are developed for each control decision and

implemented in the simulation model. The simulation model is expected to support both the development of conceptual design methodology and the development of operational control strategies of SBS/RS warehouses, to be further discussed in Chapter 4 and Chapter 5, respectively. Finally, the simulation techniques applied are viewed as potentially expandable to similar warehousing systems of larger scopes or/and applying different AS/R technologies.

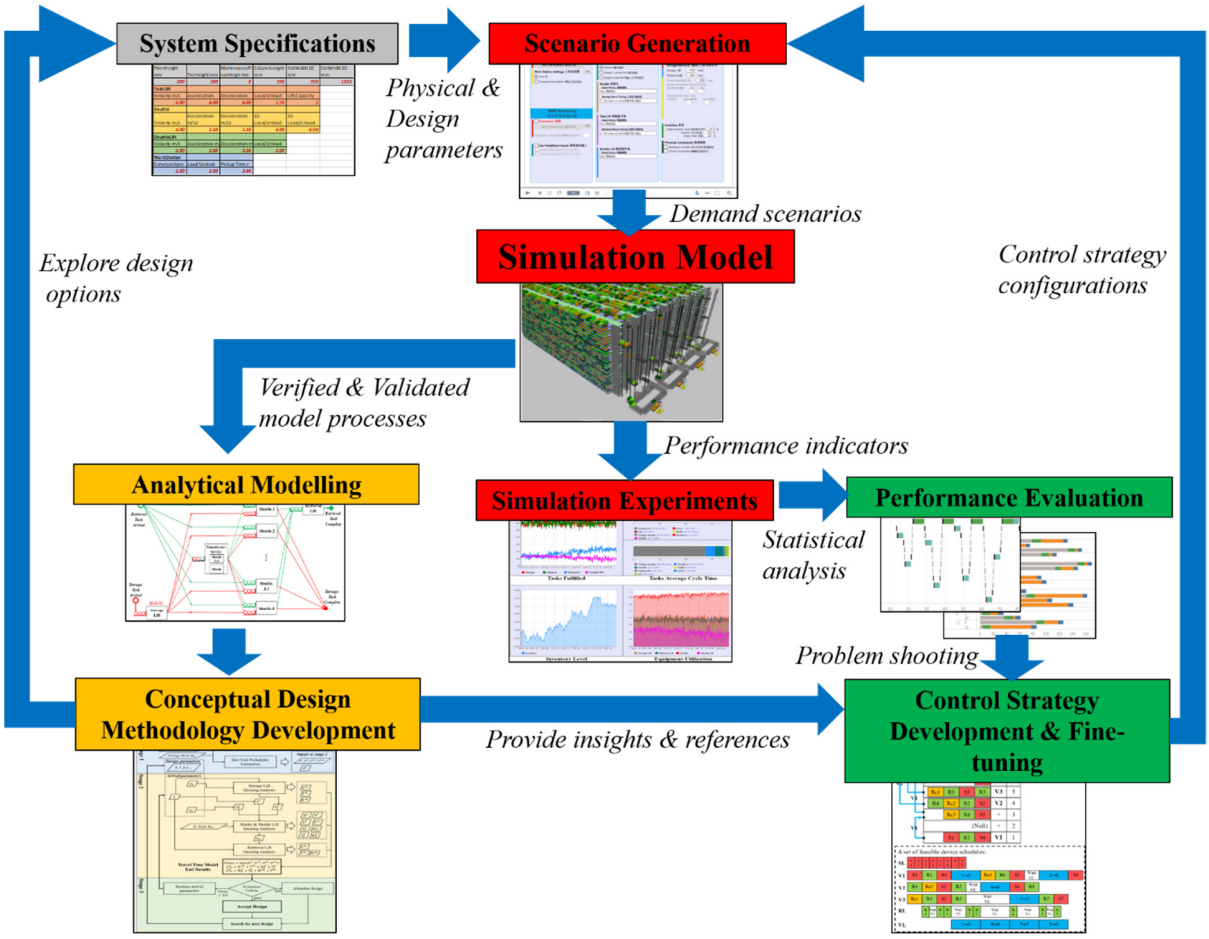

Figure 3.5 Using the Simulation Model for Conceptual Design and Control Strategy Development

Chapter 4 System Design, Configuration and Performance Analysis

4.1 Overview

In this chapter, a general analytical approach is established to support the conceptual design of Shuttle-based Storage and Retrieval Systems (SBS/RSs). A comprehensive travel time model based on queuing network analysis is developed to capture critical aspects from both the system design and operational control perspectives. Based on the travel time model, a precise and efficient three-stage iterative analytical approach is proposed. In this approach, design candidates – described by parameters including numbers of aisles, tiers, columns, rack depth, and shuttles – are evaluated and screened, in order to find the best design(s) for the given application environment described by demand variables including distributions of inter-arrival times of storage tasks and retrieval tasks as well as the inventory levels (rack utilizations). System performance under evaluation is indicated by system throughput and task responsiveness (cycle times). Various design options including tier-captive configurations and tier-to-tier configurations, multi-deep racks, and multi-unit tote lifts are accommodated in our approach. For all these design options, work patterns of different types of S/R devices and their correlations are analyzed under a general framework. Moreover, effects of operational control strategies including storage assignment policies and device scheduling policies are incorporated in our approach. Finally, the travel time model is validated by Monte-Carlo experiments with an animated, data-driven and data-generated simulation model that we developed (described in Chapter 3). The most complicated system configurations researched in this chapter are aisles with 2-deep racks (thus tote relocation operations are considered), 2-unit tote lifts (thus tours with different sizes are considered), Closest Open Location policy for storage assignment, and Dual Cycle policy for shuttle scheduling. Various demand levels, rack sizes, rack aspect ratios and shuttle deployment rates are examined

in the validation experiment. Even for such complicated systems, the validation results are still satisfactory: for tier-captive configurations, the analytical approach provides 98.7% and 98.0% precision on average with the throughput estimates and task cycle time estimates, respectively; for tier-to-tier configurations, the analytical approach provides 95.1% and 92.2% precision on average with the throughput estimates and task cycle time estimates, respectively.

4.2 Key Factors in SBS/RS Performance Evaluation

During the conceptual design phase of an SBS/RS-based warehousing system, the designer needs to determine the system configuration, which is characterized by the number of aisles, number of tiers, depths, as well as the number of shuttles to be deployed to each aisle. The target of conceptual design phase is to find the most cost-effective system configuration that satisfies the performance requirements under the expected operational environment, while also meeting other constraints like capacity requirements, layout requirements, maximum height, etc.

The performance of an SBS/RS configuration is measured by its efficiency serving storage tasks and retrieval tasks. It is usually indicated by **throughput**, which is the maximum service rate of the tasks, and **responsiveness**, which is the cycle time of the tasks. Compared to the throughput indicator, the task responsiveness might be a secondary concern in warehousing applications. However, in production applications, timing could be critical in retrieving inventory products to the production line.

An SBS/RS may have multiple aisles, interfacing with external systems like conveyor networks, pickup and/or packing workstations, etc. As each aisle of the SBS/RS has its own set of L/U devices that do not directly interfere with those in other aisles, each aisle can be analyzed independently, and the overall system performance can be viewed as an aggregation of each aisle's performance under reasonable assumptions and/or approximations. As such, in this chapter, the research mainly focuses on analytical modeling of a single SBS/RS aisle.

4.2.1 Demand Scenario Characteristics

From the perspective of an individual SBS/RS aisle in a larger system, the aisle fulfills external demands by performing storage tasks and retrieval tasks. Thus, to evaluate an SBS/RS configuration for a target application environment, the demand scenario should be primarily

interpreted in terms of arrival patterns of storage/retrieval tasks. In our analytical modeling approach, such patterns are presented as random distributions of inter-arrival times. The task arrivals are not necessarily Poisson (exponential inter-arrival time).

The range of total inventory levels is another important part of a demand scenario. Based on this, a target capacity is set to narrow the search space for configuration candidates. Also, the rack utilization range of each aisle is assumed, which is one significant input in estimating system's throughput and responsive performance.

Moreover, depending on the application environment, different demand scenarios may also have different requirements for the throughput and responsiveness of the system. Such requirements could be either interpreted as rigid constraints (e.g. minimum retrieval time requirement in the production line case) or as a weighed contribution to an overall cost-based objective function.

Finally, SKU (stock keeping unit)-level characteristics may also be abstracted as model inputs to improve evaluation precision. For example, with a multi-deep rack aisle, SKU assortment could significantly impact the shuttles' relocation workload. Low SKU assortment means totes containing same SKU types are more likely stored in neighboring deep slots, thus fewer relocation tasks are needed and the average service time is reduced. In this case, the system throughput will be underestimated if assuming unique SKU in each tote.

4.2.2 Rack Dimensions and Device Characteristics

In an SBS/RS aisle, horizontal and vertical L/U devices interact to perform storage and retrieval tasks, and each type of the devices could become the potential bottleneck to the system's throughput. For tote lifts (the storage lift and the retrieval lift), their service rates are primarily limited by the height of the rack – the higher the rack, the longer the average travel time per task. Likewise, service rate of each shuttle is primarily limited by the length of the rack. However, the overall service rate of all shuttles is determined by both the length of the rack and the number of shuttles in the rack. In a tier-captive configuration, the number of shuttles is equal to the number of rack tiers. In a tier-to-tier configuration, the time needed for transferring shuttles between tiers must be considered when estimating/computing the total service time. The throughput of the aisle could be viewed as its service rate when either the shuttles or the lifts have reached a relatively

high utilization rate (e.g., 90%). Thus, the rack dimensions (including the number of shuttles deployed) determines the upper limit of the aisle's throughput.

Device performance characteristics, including velocities and load/unload durations of each type of the devices, are the basis of travel time computations. In this research, device accelerations/decelerations are also considered. Also, tier gaps and row gaps are not necessarily assumed to be uniform, which means the model can accommodate to irregular rack designs in which distances between adjacent tiers and distances between adjacent columns (i.e., "gaps") vary. For multi-deep racks, the shuttle L/U time from/to different deep slots are handled differently.

4.2.3 Control Strategy

Control strategy is significant to the SBS/RS's performance. To explore the true potential of a candidate system configuration, the expected effectiveness of the control strategy should be taken into consideration. In a "basic" SBS/RS aisle, shuttles are tier-captive thus no tier-transfer operations, tote lifts carry one tote in each tour, and the rack is single deep. Control is more complicated with tier-to-tier configurations or/and with tote lifts capable of carrying multiple totes in each tour, or/and with multi-deep racks in which relocation operations are needed to retrieve from deeper slots blocked by neighboring shallower slots. In general, the control strategy of an SBS/RS is categorized in terms of three aspects:

1) ***Dispatching***, which creates retrieval tasks according to higher-level customer demands or production requests. This control aspect is mostly depending on SKU-level characteristics of the application environment. Decisions need to be made when the target SKU type exists in multiple storage locations. Also, specific precedence constraints or/and due-date requirements may apply depending on the demand pattern. As this chapter mainly focus on developing a general analytical approach, the dispatching aspect is not discussed in depth here.
2) ***Storage assignment***, which creates storage tasks and determines target storage locations for the totes to be stored. Storage assignment is critical to system performance as it significantly impacts the device travel times. For example, in a long rack where shuttle service rate is the system's bottleneck, a Closest Open Location (COL) type storage policy is in general more advantageous than Pure Random Storage (PRS) policy in reducing

shuttle travel time and thus improving both system throughput and task responsiveness. This is especially true when the average rack utilization is relatively low.

3) ***Device scheduling***, which determines the operation sequences for all types of devices in order to complete tasks. For example, just like in crane-based AS/RS, there is concept of Dual Cycle (DC) operations in SBS/RS: as the input/output locations on each tier are on the same side, it could be more efficient for each shuttle to serve the tasks in a retrieval-storage-retrieval-storage…manner so that to reduce travel time. The complication here relative to a crane-based AS/RS is that multiple independent devices (lifts, shuttles) are required for each task rather than a single crane.

4.2.4 Capital Expenditures

The capital expenditures of each SBS/RS aisle consists of three major elements: the rack costs, the L/U device costs, and the software costs. The rack costs are mainly determined by rack dimensions. The tote lifts are aisle-captive thus mainly determined by the number of aisles. The issue is more complicated for the shuttles. With tier-captive configurations, the number of shuttles to be purchased for an aisle is equal to the number of tiers. With tier-to-tier configurations, the decision maker could expand the number of shuttles later if the demand grows, while extra initial investment needs to be made for the shuttle lift (which is aisle-captive). The computation of operational costs associated with the SBS/RS, including operator costs, energy consumption and maintenance costs, will not be considered in this chapter.

4.3 Analytical Approach Framework

Considering the variation of demand scenarios, the flexibility of design configurations, as well as the complexity of device service patterns and the corresponding control strategies, it is not practical to establish an analytical model for SBS/RS which is both generalized and also precise. Having the analytical model alone is not adequate in control strategy development aiming at improving system performance of particular designs under particular demand scenarios. On the other hand, our simulation model is capable of providing adequate precision in evaluating particular scenarios/designs/strategies. However, in the conceptual design phase, simulation modeling alone may not be practical considering the large search space of candidate design configurations. For each single aisle, number of tiers, number of columns, rack depth, and number

of vehicles (as in tier-to-tier system) are all decision variables, which implies a four-dimension search space.

Hence, we propose a development procedure that combines an analytical approach and a simulation approach. A travel time model based on queuing analysis is developed to provide performance estimates for candidate designs. The first indicator is the system's throughput, which is the system's maximum sustainable service rate – in our approach, it is viewed as the task arrival rates (for both storage and retrieval tasks, which are assumed equal) when the utilization of one of the device types reaches a high level (e.g. 90%). The second indicator is the system's responsiveness – in our approach, it is described by the average cycle time for each task type (storage, retrieval), where the cycle time of a task is defined as its total waiting time plus total service time of all its service stages in the system. Note that the average cycle times are estimated separately for the two task types, since the responsiveness requirements for storage and retrieval tasks could be very different depending on the application areas. Validated by Monte-Carlo simulation experiments, the travel time model provides good estimates of the system's performance indicators under various assumptions regarding demand, design, and control strategy. Specifically, the characteristics of the travel time model can be described as follows:

1. The aisle is viewed as a multi-stage queuing network with two independent arrival processes (storage tasks and retrieval tasks), each has arbitrary mean and variance of its inter-arrival time distribution;
2. Applies to single aisle with arbitrary rack size parameters (number of tiers/columns, tier height, column width, etc.) and can accommodate irregular sized racks;
3. Accommodates both PRS and COL in storage policy assumptions;
4. Considers both DC and SC operations in shuttle service, where their occurrence probabilities θ^D and $1-\theta^D$ are estimated analytically;
5. Accommodates both 1-deep racks and 2-deep racks – for the latter case, occurrence probability of relocation tasks θ^R is estimated analytically, and the relocation process is incorporated in analytical computation;
6. Accommodates both tier-captive systems and tier-to-tier systems – for the latter case, occurrence probability of tier-transfer tasks θ^T is estimated analytically, and shuttle lift service and tier-transfer process are incorporated in analytical computation;

7. Accommodates multi-unit tote lifts;
8. Incorporates device acceleration and deceleration effects.

Based on various demand scenario assumptions, the analytical approach iteratively and systematically searches for design configuration candidates that are expected to satisfy the demand/space/budget requirements and estimates the candidates' key performance indicators obtained from the travel time model. The search scope is thus narrowed down significantly for further control strategy development using simulation. The results from the travel time model can also be viewed as baselines of system performance for the decision maker to explore further design options (e.g. improve device travel speed). Note that the design and configuration of encompassing systems, e.g. conveyor network and work stations, are not discussed here. Finally, the analytical approach can also be applied during the operational phase, for example, the estimation results could be used as baselines or even inputs for control strategy development.

4.3.1 Travel Time Model Formulation

Table 4.1 lists the notation for the basic system elements. In the following sections, this notation will be used in combinations. For example, T_S^V indicates the average shuttle service time for storage tasks, and $[S - TT - Re - R]$ indicates a service sequence where a shuttle completes a storage task (S), is then transferred to another tier (TT), relocates a blocker tote (Re) and finally serves a retrieval task (R). Table 4.2 shows the input parameters and basic assumptions for the travel time model. The aisle's rack capacity (total number of slots) can be computed as $X \times Y \times Z$, where X, Y, Z are the numbers of tiers, columns, and slots per column, respectively. Each slot can be mapped as a three-dimensional index $[x, y, z]$, where $x \in [1, X], y \in [1, Y], z \in [1, Z]$. The capacity requirement is the primary concern of warehousing system design, which is usually based on estimates of SKU-level demand scenarios – which is determined by SKU-level characteristics including the number of items per tote, demands, turn-over rates and seasonality, etc. The rack utilization of an SBS/RS aisle at a particular time instance is denoted as ρ and defined as the number of occupied slots (number of totes in inventory) divided by the total number of rack slots. During system operation, the rack utilization ρ of an SBS/RS aisle varies over time as it is subject to the inventory level of the warehouse – generally speaking, the higher the inventory level, the more totes in each aisle's rack, and thus the higher ρ. In addition, ρ varies over time based on the arrival patterns of storage and retrieval requests. Such patterns are specific to the configurations

and performances of higher level management decisions/systems (e.g. Material Requirement Planning, Manufacturing Execution Systems) as well as the upstream/downstream systems connected to the SBS/RS (e.g. conveyor networks).

Table 4.1 System Elements Notation

Notation	Definition	Notation	Definition
S	Storage	SL	Storage Lift
R	Retrieval	RL	Retrieval Lift
Re	Relocation	TL	Tote Lift (general, either SL or RL)
TT	Tier-transfer	VL	Shuttle Lift (Vehicle Lift)
U	Device utilization	W	Waiting Time
T	Service Time	CT	Cycle Time

Task Arrival Assumptions

The inflows and outflows of totes in the aisle's rack storage are determined by the arrivals of storage and retrieval tasks. As introduced in the previous chapters, it is defined that each task (either type) is related to a single tote. Each tote is assumed to contain a unique SKU, thus each storage task is only related to a single and unique tote. The task arrival patterns of the storage and retrieval flows are important to the travel time computations. First of all, the system's throughput is defined as the maximum sustainable task service rate. Secondly, the system's responsiveness performance (in terms of task cycle times) considers of both the service times by the S/R devices and also the waiting times for device services, while the latter is largely affected by the variances of task inter-arrival times according to queuing theory. Finally, with specific control policies applied, system performance is subject to the dynamics of rack utilization (number of occupied slots divided by total slots, denoted as ρ) – which is determined by the correlations between storage arrivals and retrieval arrivals. For example, Closest-Open-Location policy (COL) is a classic storage assignment approach studied in this research which always assign storage totes to closest available slots to the I/O buffers on each tier – when COL is applied, higher rack utilization on each tier means longer expected shuttle service times for both storage tasks and retrieval tasks on this tier.

Essentially, the arrivals of retrieval and storage tasks to the SBS/RS, as well as the rack utilization dynamics, are subject to the SKUs' demand and replenishment patterns at the warehouse system level. In other words, the task arrival patterns are determined by higher-level inventory control decisions external to the SBS/RS. As discussed in the previous chapter, it is not practical to explore all of such complexities with regard to task arrival patterns, not only because they are specific to the application environment but also because they may not be predicable at the design conceptualization phase. In this chapter, we are developing a generalized analytical methodology to provide performance estimation of the SBS/RS, based on which design candidates are evaluated for conceptual design purpose. For that purpose, proper simplification assumptions need to be made so that to decouple the aforementioned external complexities and the SBS/RS design and configuration evaluation. In the analytical approach, the total rack capacity requirement, the inter-arrival time (IAT) of tasks, as well as the dynamics of rack utilization, are assumed as inputs given beforehand. The task arrival patterns are formulated as general distributions of the inter-arrival times of both task type (storage and retrieval) to the aisle, defined as the arrival rate λ (the inverse of average inter-arrival time) for both task types and standard deviations σ_S and σ_R for storages and retrievals, respectively. The time-varying rack utilization of the aisle is described as a probability distribution function $I(\rho)$ to present the correlations between storage arrivals and retrieval arrivals. The determination of $I(\rho)$ will be further introduced in Section 4.4. The task arrival assumptions (IATs and $I(\rho)$) are the basis for the queueing-network based analysis which is the core part of the travel time model – to be discussed in detail in Section 4.5. Finally, in this analytical approach it is assumed that no precedence constraints exist between storage or retrieval tasks (although precedence may be common for systems in production line applications, we view it as a control issue to be addressed with later using simulation-based control strategy development).

Table 4.2 Input Parameters and assumptions of an SBS/RS Aisle

Demand Scenario	
Notation	**Definition**
λ	Arrival rate of storage tasks = Arrival rate of retrieval tasks
σ_S	Standard deviation of inter-arrival time of storage tasks
σ_R	Standard deviation of inter-arrival time of retrieval tasks
$I(\rho)$	Probability distribution function of the aisle's rack utilization

Task correlations: Inter-arrival times (IATs) of storage and retrieval tasks are assumed to be independent random distributions $(1/\lambda, \sigma_S)$ and $(1/\lambda, \sigma_R)$, respectively. Each storage/retrieval task is related to one single and unique tote, and not subject to precedence constraints with any other storage/retrieval task(s). The correlations between storage arrivals and retrieval arrivals are abstractly modeled using probability distribution $I(\rho)$.	
Design and Configuration	
Notation	**Definition**
X	Number of tiers
Y	Number of columns in each tier
Z	Number of slot locations in each column ($= depth \times 2$)
J	Number of shuttles $\begin{cases} J = X, tier - captive\ configuration \\ J < X, tier - to - tier\ configuration \end{cases}$
K^{TL}	Capacity of tote lift
$\tau^{TL}_{x1,\ x2}$	Tote lift travel time from current tier $x1$ to target tier $x2$, where $x1,\ x2 \in [0,\ X]$ (0 = Load point of storage lift / Unload point of retrieval lift)
$\tau^{VL}_{x1,\ x2}$	(Tier-to-tier configuration) Shuttle lift travel time from current tier $x1$ to target tier $x2$, where $x1,\ x2 \in [1,\ X]$
$\tau^{V}_{y1,\ y2}$	Shuttle travel time from current column $y1$ to target column $y2$, where $y1,\ y2 \in [0,\ Y]$ (0 = Load/unload point of I/O buffer on each tier)
ω^{TL}	Tote lift load/unload time per tote
ω^{VL}	Shuttle lift load/unload time
ω^{V}_{0}	Shuttle load/unload time from/to I/O buffer
ω^{V}_{z}	Shuttle load/unload time from/to a slot of location $z \in [1,\ Z]$
System boundaries: The load point of the storage lift (thus the entry point of the system) and the unload point of the retrieval lift (thus the exit point of the system) are both assumed to be unique	
Operational Control	
Notation	**Definition**
M_{SA}	Storage assignment mode within tier. $M_{SA} \in [COL, PUR]$
M_{DC}	Shuttle dual-cycle mode within tier. $M_{DC} \in [True, False]$
Scheduling policy: Each device is assumed to serve tasks available to it in first-come-first-serve (FCFS) pattern (additional rule for shuttles when $M_{DC} = True$)	
Dwell-point policy: Each device stays at the point-of-service-completion (POSC) when it becomes idle	
Relocation policy:	

With multi-depth rack, when a relocation service is required in order to retrieve a blocked tote, the target slot of the blocker tote to be relocated is selected based on the same criterion as the storage assignment policy
Tier-to-tier policy: In tier-to-tier configuration, a tier-transfer service is started only when the following conditions are true: 1. There exists at least one task whose target tier currently has no shuttle 2. The shuttle lift is idle 3. There exists at least one shuttle which is idle and there is no task can be started on its current tier

Table 4.3 Computational Variables of an SBS/RS Aisle

Notation	Definition
$P^{S}_{x,y,z}$	Probability a storage task is to slot $[x, y, z]$
$P^{R}_{x,y,z}$	Probability a retrieved task is from slot $[x, y, z]$
$P^{ReI}_{x,y,z}$	Probability relocation is performed to slot $[x, y, z]$ in a retrieval task
$P^{ReO}_{x,y,z}$	Probability relocation is performed from slot $[x, y, z]$ in a retrieval task
θ^{D}	Ratio of occurrences of dual cycles in in-tier shuttle tasks (excl. tier-transfer tasks)
θ^{R}	Ratio of occurrences of relocations over total retrieval tasks
θ^{T}	Ratio of occurrences of tier-transfer tasks over total tasks

Devices Travel Time Assumptions

Devices travel between discrete *nodes* to perform tasks – the nodes are vertical tier locations for the lifts, and horizontal column locations for the shuttles. In our model, the device travel times between each pair of nodes are precomputed based on device velocity characteristics (maximum speed, acceleration/deceleration, etc.) and the distance between nodes, thus constructing a travel-time matrix for each device type. In regular-shape racks, the gaps between nodes are uniform – if devices are assumed with constant velocity or constant acceleration/deceleration, the travel times could also be presented in simpler, continuous forms (as illustrated in Section 3.2.1). However, here we choose to present travel times in such discrete, matrix forms and use them as direct inputs to the model, so that allowing flexibility in accommodating for irregular-shape racks and cases in which the devices have complex

acceleration/deceleration patterns. For the same reason, rack gaps and device velocity characteristics are not included as direct inputs to the model.

Control Policies Assumptions

The storage assignment policy determines the target slot for each storage task. It is assumed that the target tier for each storage task is selected randomly. However, after the tier is selected, the slot is selected either randomly (PRS) or according to proximity to I/O buffers (COL) – this option is an input parameter to the model. In this chapter, both PRS and COL are analyzed based on some simplification assumptions of the SKU-level characteristics (e.g. the turnover rate of each SKU type, the quantities of SKUs in each tote, the demand correlations of SKU types…). The COL policy analyzed in this chapter assumes the ideal case that the probabilities all totes being retrieved are equal regardless of SKU-level characteristics, thus each incoming tote can be stored to any available slot that minimizes device service times. The PRS policy analyzed in this chapter resembles another extreme scenario where the SKU-level characteristics are totally unknown/unpredictable, and the slot of each incoming tote is predefined from all available slots with equal probabilities. Thus, the storage policy parameter here should be chosen according to the expected application environment of the system. Finally, with multi-deep rack, it is assumed that shallower slots are always prioritized over deeper slots. The detailed mechanisms and analysis will be presented in Section 4.4.

In general, each device is assumed to serve tasks available to it in first-come-first-serve (FCFS) pattern. For a storage task, it is not available to a shuttle until the storage lift unloads its tote to the input buffer of the target tier. Similarly, a retrieval task is not available to the retrieval lift until a shuttle unloads its tote to the output buffer of its current tier. In addition, shuttles can be assumed to apply dual-cycle (DC) policy or not – this option is an input parameter to the model. DC policy is expected to reduce shuttle travel time. If DC is applied, when there is at least one storage task and one retrieval task available to a shuttle, it will prioritize task(s) of the different type from its previous task. For example, if the previous task is retrieval (which means the shuttle's current position is tier I/O), it will select a storage task to proceed with if possible – if multiple storage tasks are available, FCFS rule is applied.

In our travel-time model, COL policy is not applied in selecting target tiers (which means slots in tiers closer to system entry/exit are not prioritized). This is because there can exist at most

one shuttle on each tier, simply prioritizing particular tiers will make some shuttles overloaded while others poorly utilized. A more advanced storage policy that dynamically considers both travel times and device workloads would be preferable in control strategy development. However, it will not be explored in this analytical approach – as the travel-time model is developed for higher-level conceptual design purpose. Similarly, as illustrated in Table 4.2, relatively simple assumptions are made for other control policies involving device scheduling, dwell-point, relocation service and tier-transfer service. However, it is noticeable that such assumptions are only "simpler" comparing to control strategy development in the later phase, but still with sufficient precision in estimating baseline performance of the design configurations.

4.3.2 Queuing Analysis

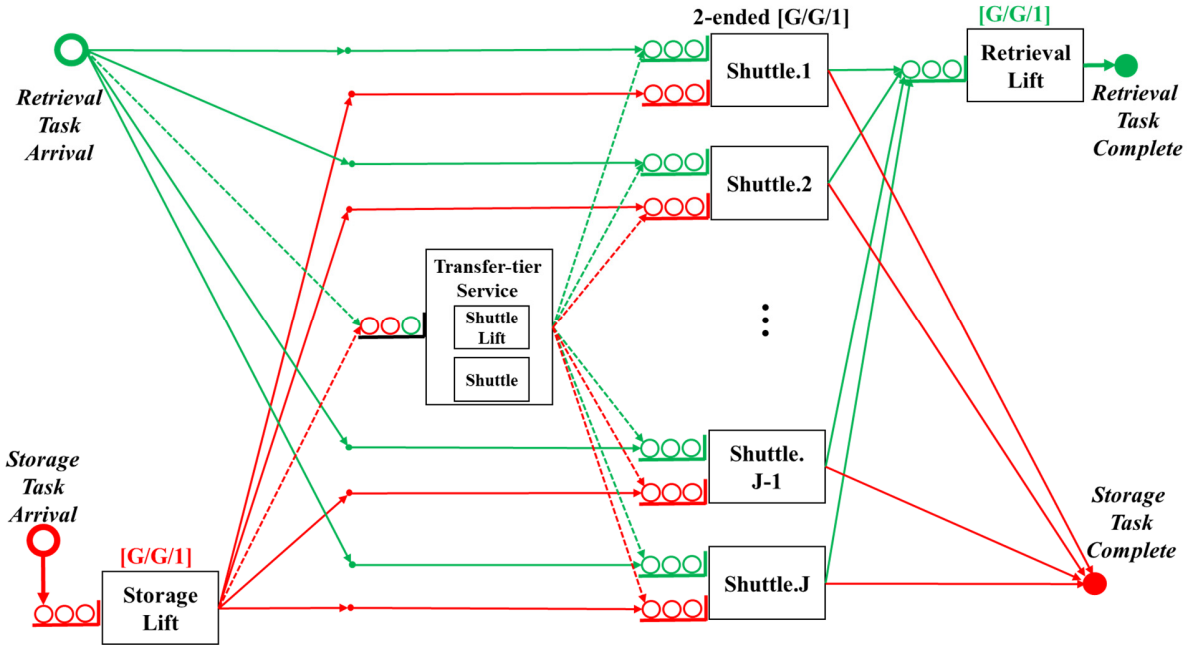

Figure 4.1 Queuing System Presentation of a single SBS/RS aisle

Although different types of devices each take care of a section of storage/retrieval tasks, the operations of those devices are not fully independent, and thus the computation of the system's actual throughput and responsiveness characteristics is complicated by these dependencies and interactions. Based on demand/design/control assumptions made in Section 4.3.1, the problem formulated previously can be presented in a queuing network form as illustrated in Figure 4.1. The

system is essentially a tandem queuing system with two external arrival processes, and both are served in two-stage patterns.

The storage lift or retrieval lift only serve a single task type. Although the storage lift and retrieval lift may carry multiple totes in each tour, they are formulated as G/G/1 queuing systems, based on approximations of the equivalent task service and waiting times with respect to the occurrence probabilities of different tour sizes. The tote lifts are not formulated as G/G/c instead for the following reasons: firstly, the service times of tasks in the same tour only partially overlap; secondly, the service times of a task is dependent to those of other task(s) in the same tour; at last, the size of a tour could between one and the lift capacity, but once the tote lift starts moving to unload the first task, no further tasks can be added to the tour even if the tour is not full. Detailed queuing analyzes of the tote lift services will be presented in Section 4.5.1.

On the other hand, each shuttle serves both types of tasks. For in-tier task services, each shuttle can be viewed as an independent, double-ended G/G/1 queuing system, based on approximations of the equivalent task service and waiting times with respect to the interleaving characteristics imposed by the DC policy as well as the relocation characteristics with multi-deep racks. For tier-transfer task services, the situation is further complicated because the tier-transfer tasks search for an idle shuttle for service. The shuttle lift is paired with an idle shuttle to perform a tier-transfer task, thus imposing the entire system additional characteristics which is similar to a G/G/c queuing system, and the equivalent service times are approximated based on additional factors including the number of shuttles, the number of tiers, the occurrence probabilities of tier-transfer tasks, etc. Detailed queuing analyzes of the shuttle services will be presented in Section 4.5.2.

The throughput of an SBS/RS aisle is defined as the maximum task service rate of the aisle, which is determined by the service rates of all device types: shuttle, storage tote lift, retrieval tote lift, and shuttle lift. In other words, any device type can become the bottleneck of the entire system when its utilization rate is too high, even if all the other device types are poorly utilized. This also brings insights to system design: for example, in a tier-to-tier configuration, adding more shuttles will not be helpful if the storage lift is expected to be the bottleneck of a rack configuration (for a given demand scenario), and the right approach might be choosing a vertically shorter while horizontally longer rack configuration instead. Task responsiveness is measured by task cycle time,

which is defined as the sum of service times plus sum of waiting times in all service stages. Because responsiveness requirements of storage tasks and retrieval tasks are usually of different importance for a warehousing system, cycle times of different task types are analyzed separately in our model. Effectiveness of control policies (storage assignment, shuttle dual-cycle, etc.) under different demand scenarios and design configurations is expected to be significant importance to both throughput and responsiveness performance, thus need to be evaluated carefully from conceptual design phase. For example: larger portion of shuttle dual-cycles and fewer occurrence of relocations are both expected to improve average service rate of shuttles; considering the tier-transfer service pattern in tier-to-tier configuration, the occurrence frequency of such service is expected to significantly affect the overall system performance. We attempt to abstract these factors to improve the validity and precision of our travel time model.

4.3.3 Iterative Searching Procedure

As introduced earlier, the service patterns of SBS/RS are complicated due to various factors including device interactions, storage policies, DC policies, relocation, tier-transfer, etc. In order to model the problem in an appropriate form for queuing analysis, it is critical to identify the important factors to system performance, describe them in appropriate mathematical formats, and integrate them into the model formulations. In

Table 4.3, a set of important computational variables critical to travel time computation are identified. In our analytical approach, based on the input parameters and assumptions introduced previously, these variables are computed step-by-step and used in queuing analysis. Figure 4.2 gives an overview of the analysis procedure based on the travel time model. For each candidate design, the slot visit probabilities $P^S, P^R, P^{RelI}, P^{ReO}$ and relocation ratio θ^R are estimated in Stage 1 given the storage mode as well as the assumptions for the rack utilization (inventory level). DC ratio θ^D and tier-transfer ratio θ^T are estimated during the queuing analysis performed in Stage 2, given the demand scenario (arrival parameters) as well as the results obtained in Stage 1. Finally, the travel time model results indicating the candidate's expected throughput and responsiveness performances are evaluated in Stage 3. If the candidate is underutilized according to the evaluation criteria (e.g. the maximum utilization among all devices U^{max} is less than 90%, which means the candidate can potentially handle larger demands), then Stage 2 is iterated with larger arrival parameters (for example, in our approach the new arrival rate λ^{new} is determined as

$\min\left[\lambda^{old}\left(\frac{0.9}{U^{max}}-1\right), 1(per\ hour)\right]$, and $\sigma^{new} = \sigma^{old}\frac{\lambda^{new}}{\lambda^{old}}$). Otherwise, the candidate is either accepted or abandoned according to the evaluation criteria. Finally, the design space is explored based on previous results and the next candidate is selected for a new analysis iteration starting from Stage 1.

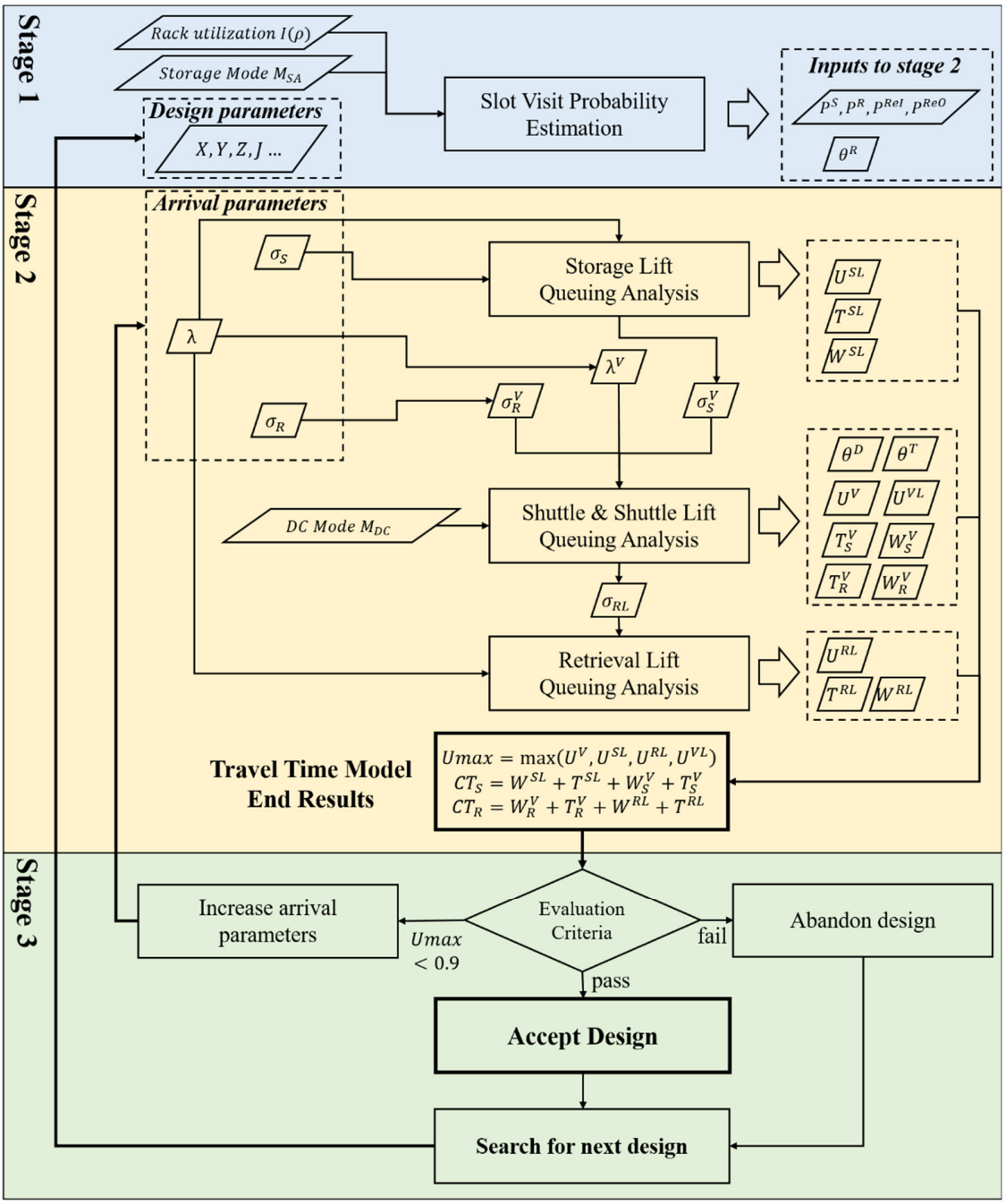

Figure 4.2 Iterative analysis procedure based on the travel time model

In this research, the iterative analysis procedure proposed in this chapter is packaged as an independent executable analysis program coded in Java. The analysis program takes external input data including the system's physical parameters and design specifications described in Excel spreadsheets.

4.4 Slot Task Visit Probabilities Estimation

This section introduces the first stage of the analysis procedure illustrated in Figure 4.2. For storage assignment policies, both Pure-Random-Storage (PRS) and Closest-Open-Location (COL) are analyzed. Also, both 1-deep racks and 2-deep racks are considered for both storage policies, thus constructing four different cases in total. In each case, the visiting probabilities of each slot location by storage tasks, retrieval tasks, and relocation operations (in 2-deep racks) are estimated. These probabilities are the basis for the service time modeling based on queuing analysis introduced in the next section. It is noticeable that these probabilities are considered independently from the devices' performances (velocities, L/U times, etc.) and independently from the demand levels (thus the device utilizations) as well.

A notation $N_{x,y,z}$ is used to define the occupancy probability of slot $[x, y, z]$. Based on rack design, rack utilization and storage policy mode, a matrix N is formed to present the slot states in the rack. Task visit probability matrices $P^S, P^R, P^{RelI}, P^{ReO}$ as well as the relocation ratio θ^R are computed based on the matrix N and used as inputs to the next stage of the analytical approach.

4.4.1 Determination of Rack Utilization Probability Distribution

In practical warehousing operations, both the demand rates and replenishment rates of the SKUs are usually inventory dependent, thus correlations exist between storage and retrieval arrivals. As introduced previously, the development of rack utilization probability distribution $I(\rho)$ is an approximation approach for such correlations of the demand scenarios of evaluation interests. The $I(\rho)$ distribution is essentially an assumption to the travel time model, where $\rho \in (0,1), \forall \rho$ and $\int_0^1 I(\rho) = 1$. The $I(\rho)$ distribution is assumed as time independent in each demand scenario, independent from rack shape parameters (tiers/columns/depths), independent from task IAT distributions, and independent from the storage and scheduling policies. Generally speaking,

in design conceptualization, the design candidates are evaluated under demand scenarios where the racks are properly utilized (high $\overline{\rho}$). Also, $I(\rho)$ is expected to approximate the rack utilization dynamics due to routine warehouse operations within relatively short time periods (e.g. on a daily/weekly basis) instead of approximating the seasonality of the warehouse inventory level – for the latter case, system performances are expected to be evaluated by the travel time model under different $I(\rho)$ assumptions.

One important reason that a probability function $I(\rho)$ instead of a single expected value $\overline{\rho}$ is used as model input is that, the estimates will be less precise when COL is assumed for storage assignment if only a single $\bar{\rho}$ is used in the computations introduced in this section – the visit probabilities of the slots at the further end (with respect to I/O) will be underestimated. On the other hand, our experience from numerous validation experiments is that, the development of $I(\rho)$ distribution does not require perfect precision – rough estimation of $I(\rho)$ is adequate for the travel time estimation. Take an example that a user is exploring design candidates which provide around $100k$ total rack capacity with 10 identical aisles, thus $XYZ \approx 10k$. Assume the user is interested in system performance under demand scenarios where the total inventory level of all 10 aisles (in terms of totes) varies on a daily/weekly basis according to a particular distribution $normal(80k, 5k)$ – such distributions are assumed to be obtained either based on historical data or through empirical approaches like inventory modeling. These approaches are focusing at the higher system level (warehouse or supply chain level) and thus not discussed in the scope of this research. Then, it is reasonable to assume $I(\rho)$ of each aisle to be $normal(0.8, 0.05)$ for this example since the aisles are identical – otherwise (e.g. aisle shapes are different, priority rules exist between aisles, or aisles are predetermined for different SKU types), the performance of each aisle may need to be estimated separately by the travel time model with separate $I(\rho)$ and IAT settings $(\lambda, \sigma_S, \sigma_R)$ as well. Denote ρ_x as the instantaneous utilization of tier $x \in [1 \dots X]$ (thus the number of occupied slots on this tier equals $\rho_x YZ$), and denote P_x^S and P_x^R as the probabilities that an incoming storage task / retrieval task is targeting at tier x. If not otherwise specified regarding the preferences between tiers in storage assignment, there are:

$$\begin{cases} P_x^R = \dfrac{\rho_x YZ}{\rho XYZ} = \dfrac{\rho_x}{\rho X} \\ P_x^S = \dfrac{(1-\rho_x)YZ}{(1-\rho)XYZ} = \dfrac{(1-\rho_x)}{(1-\rho)X} \end{cases}, \forall x \in [1 \dots X]$$

Denote $I_x(\rho)$ as a probability function of rack utilization on tier $x \in [1 \ldots X]$. A series of Monte Carlo simulation experiments are conducted to analyze the relationship between $I(\rho)$and $I_x(\rho)$ under different designs and demand scenarios. As illustrated in Figure 4.3, we observed that for typical designs where number of tiers are not too small (e.g. $X > 5$), when the covariance of the IAT distributions $cov_S = \lambda\sigma_S$ and $cov_R = \lambda\sigma_R$ are not too small ($\min(cov_S, cov_R) > 0.5$), it is reasonable to assume $I_x(\rho) \approx I(\rho), \forall x$.

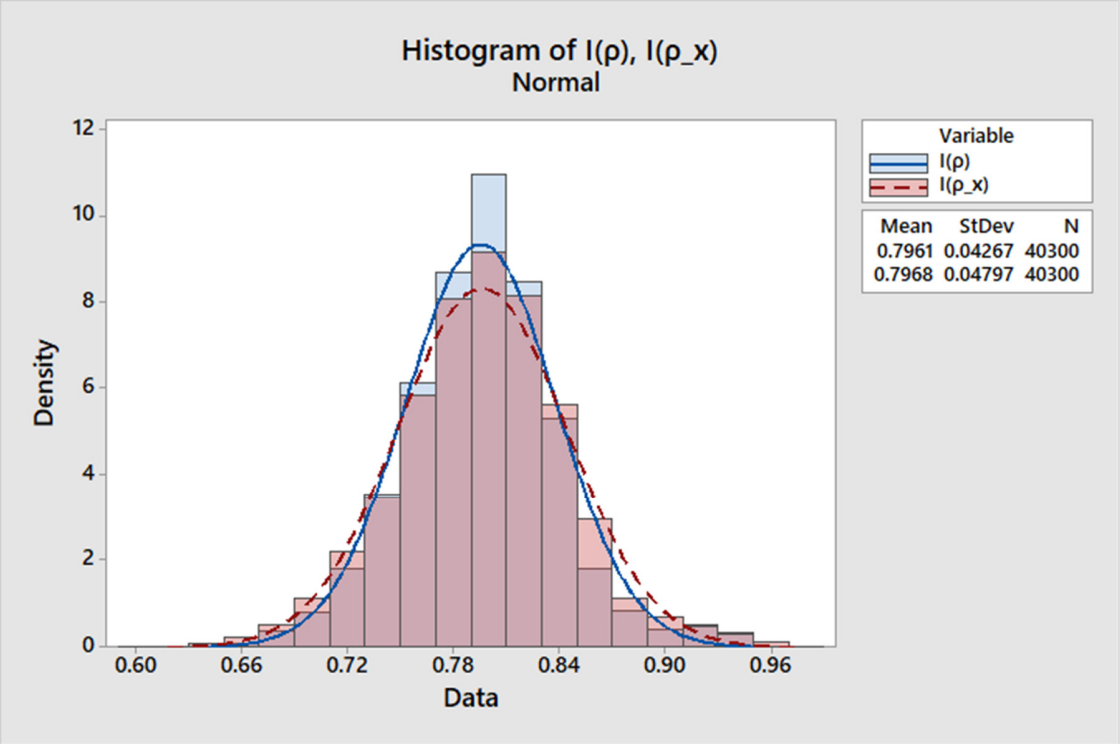

Figure 4.3 Rack utilization PDFs (Aisle vs. Single tier) of a 14-tier, 73-col, 2-deep design under $cov_S = cov_R = 1$ scenario, based on Monte-Carlo Simulation

If no historical data or empirical approaches are available for rack utilization estimation (which is very likely in the conceptual design phase), we hereby propose an alternative approach based on Monte Carlo simulation to approximate $I_x(\rho)$ of an arbitrary tier (assuming no preferences between tiers in storage assignment). Consistent with the demand scenario modeling approach introduced in Section 3.2.2, storage and retrieval arrivals are generated according to the following inputs:

1) Design parameters: X, Y, Z;
2) Demand parameters: Expected rack utilization $\overline{\rho}$ and IAT parameters: $\lambda, \sigma_S, \sigma_R$;
3) Time window: T, assuming the total storage arrivals equals the total retrieval arrivals within duration T.

Then, $I_x(\rho)$ is estimated by sampling the resulting rack utilization ρ's along the time. Details of this approximation approach is further introduced in Appendix I. Based on experimenting different combinations of design parameters, demand parameters and T (days), we observed that $I_x(\rho)$function roughly approximates a normal distribution truncated at $\rho = 0$ and $\rho = 1$, with expected value $\overline{\rho}$, and its standard deviations $\sigma_x(\rho)$ can be approximately described by the following regressing model:

$$\sigma_x(\rho) \approx [2 + \lambda(\sigma_S + \sigma_R) + 0.14T + 0.07X - 0.0003XYZ] \times 100\%$$

Finally, $I_x(\rho)$ is converted into a discrete probability distribution from its continuous form – which is either estimated based on historical data / empirical approach, or the $normal(\overline{\rho}, \sigma_x(\rho))$ distribution obtained from the above Monte Carlo approach. In the rest of this section, the discretized $I_x(\rho)$ is used as a direct input to estimate the slot task visit probabilities. The empirical approach we use is dividing $I_x(\rho)$ into equal-length intervals. When given $normal(\overline{\rho}, \sigma)$ ($\sigma_x(\rho)$ is presented as σ for reading clarity) the distribution is divided into seven intervals within range $[\rho_{min} = \max(\overline{\rho} - 3\sigma, 0), \rho_{max} = \min(\overline{\rho} + 3\sigma, 1)]$ as follows:

$$\rho_i = \rho_{min} + (i - 0.5)\frac{\rho_{max} - \rho_{min}}{7}, i \in \{1,2 \dots 7\}$$

$$I_x(\rho_i) = \frac{P\left(\rho_i - 0.5\frac{\rho_{max} - \rho_{min}}{7} < \rho < \rho_i + 0.5\frac{\rho_{max} - \rho_{min}}{7}\right)}{P(\rho_{min} < \rho < \rho_{max})}, \forall i$$

For example, with the design and demand scenario presented in Figure 4.3 Rack utilization PDFs (Aisle vs. Single tier) of a 14-tier, 73-col, 2-deep design under $cov_S = cov_R = 1$ scenario, based on Monte-Carlo Simulation we have $\overline{\rho} = 0.8$ and $\sigma = 0.0473$, and $I_x(\rho)$ is discretized as in Figure 4.4.

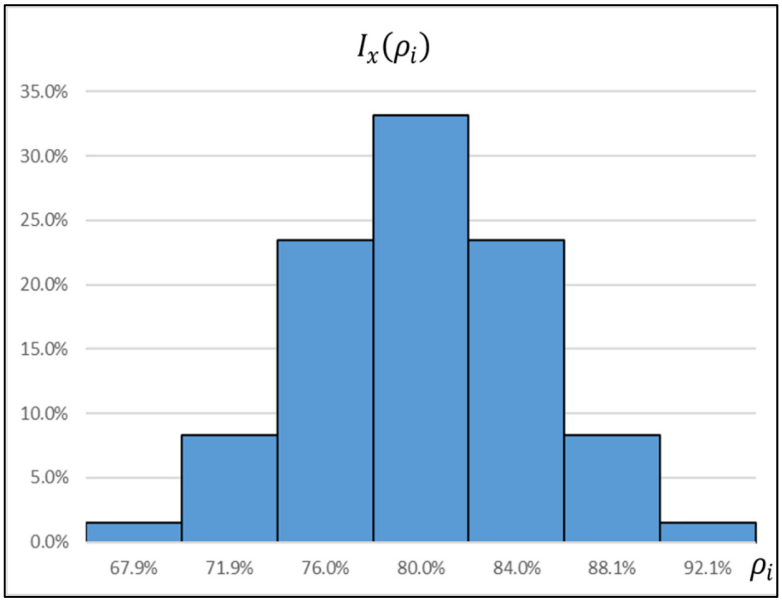

Figure 4.4 Discretized $I_x(\rho)$ of a 14-tier, 73-col, 2-deep design under $cov_S = cov_R = 1$ scenario, based on Monte-Carlo based estimation approach (T=1 week)

4.4.2 Approximations of Storage Policies

Storage policy has significant impact on the service rates of the devices, while its selection is highly dependent on the system's application environment. In our travel-time model, the storage policy of the aisle is approximated either as Pure Random Storage (PRS) mode or as Closest Open Location (COL) mode. In general, COL-type storage policy increases the occupancy rates of forward storage locations (slots closer to I/O) and thus reduces device travel times. It is likely that customized Class-based storage policies based on SKU-level analysis would perform even better for particular application environments. However, such advanced policies are expected to be explored in the control strategy development phase and thus are not discussed here. For general e-commerce warehousing systems, their storage policies can be assumed as COL in the conceptual design phase.

4.4.3 Analysis of 1-deep Rack

Figure 4.5 illustrates a single tier of a 1-deep rack design, where only one row of slots on each side of the aisle. Coordinates on the top shows slot locations, number on the bottom indicates selection priority of each slot in its tier with COL storage.

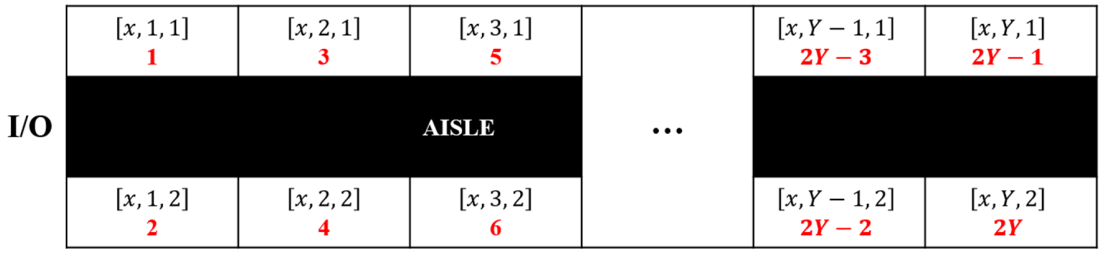

Figure 4.5 Illustration of slots on a single tier (tier x) of a 1-deep rack and storage assignment priorities under COL

Pure Random Storage (PRS) Mode

When assuming PRS storage mode, an empty slot is selected randomly for each storage task, which means the occupancy probability of each slot and the probability each slot is visited in a storage or retrieval task are both uniform. Thus, in a 1-deep rack applying PRS, when the rack utilization is ρ, for all slots there are:

$$N_{x,y,z} = \rho$$

$$P^S_{x,y,z} = P^R_{x,y,z} = \frac{1}{2XY}$$

Note that $2XY$ is the total rack capacity of a 1-deep aisle.

Closest Open Location (COL) Mode

When assuming COL storage mode, slots closer to I/O on each tier are prioritized in storage assignment. As a result, both the occupancy probabilities and task visit probabilities of these slots are expected to be higher than those further from I/O. For the clarity of presentation, let's temporarily use slot priority index (as shown in Figure 4.5) to indicate the probabilities: P^S_i and P^R_i indicate the probability a storage/retrieval task to/from a tier is targeting slot i on this tier, and N_i indicates its occupancy probability. At a particular rack utilization ρ, it can be inferred that:

$$P^S_1 = 1 - N_1$$

$$P^S_i = \left(\prod_{i'=1}^{i'<i} N_{i'}\right) \times (1 - N_i), \forall i > 1$$

As an empty slot is selected in storage assignment to its tier only when all slots with higher priorities (closer to I/O in COL) in this tier are occupied.

$$P_i^R = \frac{N_i}{2\rho Y}, \forall i$$

As the probability a slot is visited by a retrieval task to its tier equals the slot's occurrence probability divided by the total utilization rate of its tier. Because it is obvious that $P_i^S = P_i^R, \forall i$, the following equations are obtained:

$$N_1 = \frac{1}{1 + \frac{1}{2\rho Y}}$$

$$N_i = \frac{1}{1 + \frac{1}{2\rho Y \times \prod_{i'=1}^{i'<i} N_{i'}}}, \forall i > 1$$

And P_i^S and P_i^R can be computed accordingly.

Finally, based on the control assumption made in Section 4.3 that storages are assigned randomly to tiers, it can be inferred that for all slots, at a particular rack utilization ρ, we have:

$$P_{x,y,z}^S = P_{x,y,z}^R = \frac{P_i^S}{X} = \frac{P_i^R}{X}, where\ i = 2(y-1) + z$$

4.4.4 2-deep Rack and Relocation

For the purpose of improving storage space utilization, use of multi-deep racks is a popular option in automated warehouses. Multi-deep aisle design provides higher storage density compared to 1-deep aisle design. However, the horizontal L/U mechanism in multi-deep capabilities is generally more expensive because it either requires shuttles capable of accessing deeper slots (usually no more than 2-deep), or requires four-directional shuttles or additional L/U devices (such as satellite shuttles) that travel along the z-axis. In this research, only 2-deep rack design is studied for multi-deep rack designs, and shuttles are assumed as capable of reaching 2-deep positions with longer L/U times (as illustrated in Figure 4.6). In 2-deep racks, retrievals from a deep slot $[x, y, z]$ ($z \in [3,4]$) will be blocked by totes stored in its neighbor slot $[x, y, z-2]$ – in such case, relocation is needed to first move the blocker tote to another slot before retrieving the

target tote. A relocation service includes load—move—unload—move processes that incurs similar service time relative to that of a non-relocation retrieval task. Thus, minimizing the occurrence of relocation is very important for improving the system performance.

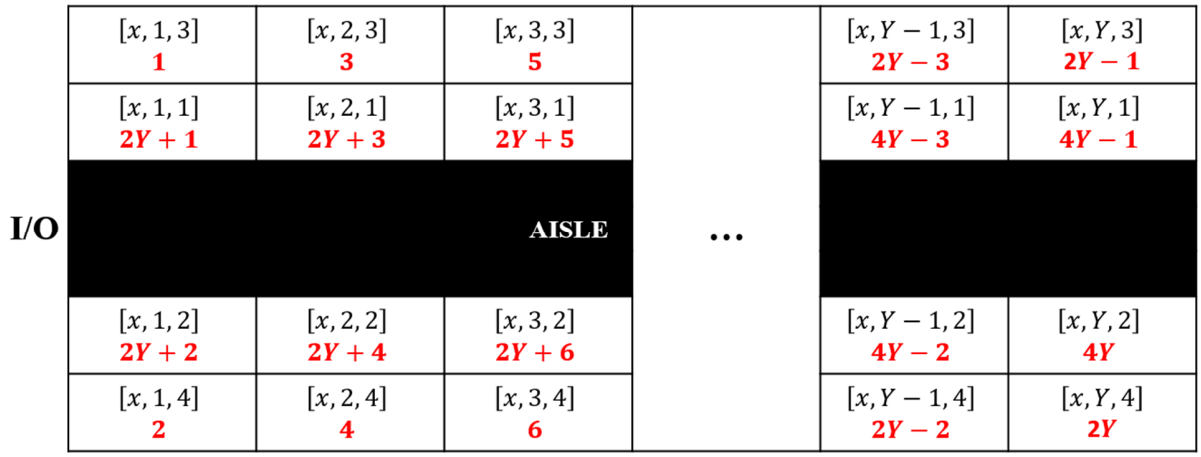

Figure 4.6 Illustration of slots on a single tier (tier x) of a 2-deep rack and storage assignment priorities under COL

Just like the previous illustration of 1-deep rack, in the 2-deep rack illustration in Figure 4.6, coordinates on the top shows slot locations, number on the bottom indicates selection priority of each slot in its tier with COL storage. In this chapter, the 2-deep slots ($z \in [3,4]$) are assumed always prioritized over 1-deep slots ($z \in [1,2]$). In each retrieval task, if relocation is needed, the target slot of the blocker tote is assumed selected based on the same priority rule. Unlike 1-deep racks where $S_{x,y,z} = R_{x,y,z}$, in 2-deep racks, the task visit probability of each slot satisfies the following equations:

$$P^{S}_{x,y,z} + P^{RelI}_{x,y,z} = P^{R}_{x,y,z} + P^{ReO}_{x,y,z}$$

Which means the sum of storage probability and relocation-to probability each slot equals the sum of retrieval probability and relocation-from probability of this slot, and:

$$\sum P^{S}_{x,y,z} = \sum P^{R}_{x,y,z} = \theta^{R} \times \sum P^{RelI}_{x,y,z} = \theta^{R} \times \sum P^{ReO}_{x,y,z}$$

Where θ^{R} is the expected relocation occurrence probability per retrieval task. In a 2-deep rack applying PRS or COL, when the rack utilization is ρ, the overall occupancy probabilities of 1-deep slots (1D) and 2-deep slots (2D) can be estimated as:

$$\begin{cases} N_{1D} = 0 \\ N_{2D} = 2\rho \end{cases}, if\ \rho < 0.5$$

$$\begin{cases} N_{1D} = 2\rho - 1 \\ N_{2D} = 1 \end{cases}, if\ 0.5 \le \rho < 1$$

The equations above indicate that when rack utilization is lower than 50%, only 2-deep slots are utilized. According to the control assumption that 2-deep slots are always prioritized over 1-deep slots, the relocation ratio θ^R when rack utilization is ρ can be estimated as the probability that a retrieval is from any 2-deep slot multiplied by the probability that any 1-deep slot is occupied, there is:

$$\theta^R = \frac{N_{2D}}{N_{1D} + N_{2D}} \times N_{1D}$$

Thus, $\theta^R(\rho)$ can be estimated as:

$$\theta^R = \begin{cases} 0, & if\ \rho < 0.5 \\ 1 - \dfrac{1}{2\rho}, & if\ 0.5 \le \rho < 1 \end{cases}$$

Pure Random Storage (PRS) Mode

When assuming PRS storage mode, an empty 2-deep slot is selected randomly for each storage task – if no empty deep slot exist, then an empty 1-deep slot is selected randomly. Thus, the slot occupancy probabilities of each slot equals the overall occupancy probability of its depth:

$$N_{x,y,z} = \begin{cases} N_{1D}, & \forall x, \forall y, z \in [1,2] \\ N_{2D}, & \forall x, \forall y, z \in [3,4] \end{cases}$$

Then, P_{1D}^S, P_{2D}^S, P_{1D}^R, P_{2D}^R, P_{1D}^{RelI}, P_{2D}^{RelI}, P_{1D}^{ReO} and P_{2D}^{ReO} are used to present the overall task visit probabilities to each depth at rack utilization ρ. As PRS is applied for storage assignment, the visit probabilities of storage tasks to each slot can be inferred as followed:

$$P_{x,y,z}^S = \begin{cases} {P_{1D}^S}/{2XY}, & \forall x, \forall y, z \in [1,2] \\ {P_{2D}^S}/{2XY}, & \forall x, \forall y, z \in [3,4] \end{cases}$$

Where $2XY$ is the total number of slots of each depth in the aisle. The other task probabilities P^R, P^{ReI} and P^{ReO} are presented in the same way. Based on the assumption that storage and relocation use the same slot-selection criterion, it can be inferred that:

$$P_{1D}^{ReI} = \theta^R \times P_{1D}^S$$

$$P_{2D}^{ReI} = \theta^R \times P_{2D}^S$$

Then, because in each relocation, the blocked tote is always in a 2-deep slot and the blocker tote is always in the neighboring 1-deep slot, it can be inferred that:

$$P_{1D}^{ReO} = \theta^R$$

$$P_{2D}^{ReO} = 0$$

Thus, based on equation $P_{x,y,z}^S + P_{x,y,z}^{ReI} = P_{x,y,z}^R + P_{x,y,z}^{ReO}$, the task visit probability equation with PRS at rack utilization ρ can be presented as:

$$\begin{cases} P_{1D}^S \times (1 + \theta^R) = P_{1D}^R + \theta^R \\ P_{2D}^S \times (1 + \theta^R) = P_{2D}^R \end{cases}$$

Finally, for each retrieval task, the visit probability of each slot is:

$$P_{1D}^R = \frac{N_{1D}}{N_{1D} + N_{2D}} = \frac{N_{1D}}{2\rho} = \theta^R$$

$$P_{2D}^R = \frac{N_{2D}}{N_{1D} + N_{2D}} = \frac{N_{2D}}{2\rho} = 1 - \theta^R$$

Thus, at a particular rack utilization ρ, the task visit probabilities to each slot can all be computed. When $\rho \geq 0.5$, there are:

$$\begin{cases} P_{1D}^R = 1 - \frac{1}{2\rho} \\ P_{1D}^S = 1 - \frac{1}{4\rho - 1} \\ P_{1D}^{ReI} = (1 - \frac{1}{2\rho}) \times (1 - \frac{1}{4\rho - 1}) \\ P_{1D}^{ReO} = 1 - \frac{1}{2\rho} \end{cases} \quad and \quad \begin{cases} P_{2D}^R = \frac{1}{2\rho} \\ P_{2D}^S = \frac{1}{4\rho - 1} \\ P_{2D}^{ReI} = (1 - \frac{1}{2\rho}) \times \frac{1}{4\rho - 1} \\ P_{2D}^{ReO} = 0 \end{cases}$$

When $\rho < 0.5$, the visit probabilities of a 2-deep rack resembles those of a 1-deep rack:

$$\begin{cases} P_{1D}^{R} = 0 \\ P_{1D}^{S} = 0 \\ P_{1D} = 0 \\ Q_{1D} = 0 \end{cases} \quad and \quad \begin{cases} P_{2D}^{R} = 1 \\ P_{2D}^{S} = 1 \\ P_{2D} = 0 \\ Q_{2D} = 0 \end{cases}$$

Finally, $P_{x,y,z}^{S}$, $P_{x,y,z}^{R}$, $P_{x,y,z}^{ReI}$ and $P_{x,y,z}^{ReO}$ at this rack utilization ρ are computed for each slot through dividing the corresponding total probability by $2XY$.

Closest Open Location (COL) Mode

With 2-deep racks, the COL rule is assumed of lower priority than the rule that 2-deep slots are always prioritized. The slot selecting priorities in such case is illustrated in the previous Figure 4.6. The task visit probabilities here are estimated in an approach similar to the previous approach for the 1-deep rack COL case. Again, for the clarity of presentation, we use P_i^S, P_i^R, P_i^{ReI} and P_i^{ReO} to indicate the task probabilities for the slot of priority i within this tier, and N_i to indicate slot occupancy probabilities – all based on the assumption that the rack is at a particular rack utilization ρ. It can be inferred that:

$$P_1^S = 1 - N_1$$

$$P_i^S = \left(\prod_{i'=1}^{i'<i} N_{i'} \right) \times (1 - N_i), \forall i > 1$$

$$P_i^R = \frac{N_i}{4\rho Y}, \forall i$$

Because the probability of relocation from a 1-deep slot equals the probability that this slot is occupied multiplied by the probability that a retrieval occurs to its neighboring 2-deep slot, there is:

$$P_i^{ReO} = \begin{cases} N_i \times P_{i-2C}^{R} & , i > 2C \\ 0 & , i \leq 2C \end{cases}$$

Because storage and relocation are assumed using the same prioritization criterion, the probability of relocation to a slot is estimated as:

$$P_i^{ReI} = P_i^S \times \theta^R$$

The equation $P_{x,y,z}^S + P_{x,y,z}^{ReI} = P_{x,y,z}^R + P_{x,y,z}^{ReO}$ can be rewritten into slot occupancy forms as followed:

$$\left(\prod_{i'=1}^{i'<i} N_{i'}\right)(1 - N_i) \times (1 + \theta^R) = \frac{N_i}{4\rho Y}, i \in [1,\ 2Y]$$

$$\left(\prod_{i'=1}^{i'<i} N_{i'}\right)(1 - N_i) \times (1 + \theta^R) = \frac{N_i \times (1 + N_{i-2C})}{4\rho Y}, i \in [2Y + 1,\ 4Y]$$

Thus, the slot occupancy probabilities are computed as:

$$N_i = \begin{cases} \dfrac{1}{1 + \dfrac{1}{4\rho Y \times (1 + \theta^R) \times \prod_{i'=1}^{i'<i} N_{i'}}} & , i \in [1,\ 2C] \\ \dfrac{1}{1 + \dfrac{1 + N_{i-2C}}{4\rho Y \times (1 + \theta^R) \times \prod_{i'=1}^{i'<i} N_{i'}}} & , i \in [2C + 1,\ 4C] \end{cases}$$

Then, S_i, R_i, P_i and Q_i can all be computed accordingly. Finally, based on the control assumption, it can be inferred that for all slots, at a particular rack utilization ρ, there are:

$$P_{x,y,z}^S = \frac{P_i^S}{X}, P_{x,y,z}^R = \frac{P_i^R}{X}, P_{x,y,z}^{ReI} = \frac{P_i^{ReI}}{X}, P_{x,y,z}^{ReO} = \frac{P_i^{ReO}}{X}$$

In which:

$$i = \begin{cases} 2(y-1) + z + 2Y & , z \in [1,2] \\ 2(y-1) + z - 2 & , z \in [3,4] \end{cases}$$

4.4.5 Final Task Visit Probability Matrices

In the previous sections, task visit probabilities are computed based on a single rack utilization value ρ. Given the discrete probability function of rack utilization $I_x(\rho)$, the final values of the matrices are approximated as the weighted averages of those computed for each ρ, thus:

$$\theta^R = \sum [\ I_x(\rho) \times \theta^R(\rho)\ ]$$

$$P^{S}_{x,y,z} = \sum [\, I_x(\rho) \times P^{S}_{x,y,z}(\rho) \,], etc$$

It is noticeable that, under the control assumption that totes are randomly assigned to rack tiers with equal probabilities for both COL and PRS cases, there is actually no need to have the x coordinate in each of the four matrices, and the probabilities can just be presented as $P^{S}_{y,z}$, $P^{R}_{y,z}$, $P^{ReI}_{y,z}$ and $P^{ReO}_{y,z}$. However, we decide to keep the current forms so that they can accommodate other control assumptions to be explored in the future.

4.4.6 Validation

Task visit probabilities and relocation ratio are critical inputs to the following travel-time computations based on queuing analysis. These inputs are assumed independent from the following computations – in other words, we assume that the task arrival rates do not impact their visit probabilities to the slots. It is not always true from operational control perspective, as it could be more efficient to apply more advanced storage/relocation policies that respond dynamically to system status. However, such simplification is consistent with the PRS and COL assumptions made previously, thus appropriate for this conceptual design phase.

To validate the task visit probability estimation approach, simulation experiments are conducted to compare the estimates to simulation results. A 2-deep, tier-captive aisle design with of 10 rows and 73 columns (thus 2,920 slots) is selected for illustration. The rack utilization is expected to be 80% and its probability function is the same as the example from Figure 4.4. PRS and COL are both examined. In the simulation experiment, 300,000 storage tasks and 300,000 retrieval tasks are created within 500 hours (excluding the warm-up period, which is 100 hours), both with exponential inter-arrival times. This means each slot is visited by about 100 storage tasks and 100 retrieval tasks on average. In addition, some simulation settings are specifically set so that to be consistent with the analytical assumptions – for example, the I/O buffer capacity is set as infinite in this experiment.

Figure 4.7 shows the task visit probability estimates from the travel time model versus the task visit occurrences from a simulation experiment (presented as ratios to total tasks) with COL storage policy. In this experiment, shuttle utilization is 78% in the COL case and 83% in the PRS case. The tote lift utilizations are about 87% in both cases. The results are aggregated for all tiers,

and aggregated for both sides of each depth. Thus, the x-axis of the figure only shows the column index. It can be observed that the analytical estimates and simulation results are very close. After a series of simulation-based validation experiments, the task visit probability estimation is viewed as a valid approach in providing inputs to the following queuing-based analysis. (Note that the task visit probabilities validated here are only intermediate results in the analytical approach. Comprehensive validation experiments will be illustrated in Section 4.6. In the following section, we will continue with the process illustrated in Figure 4.2 using the values computed in this section.

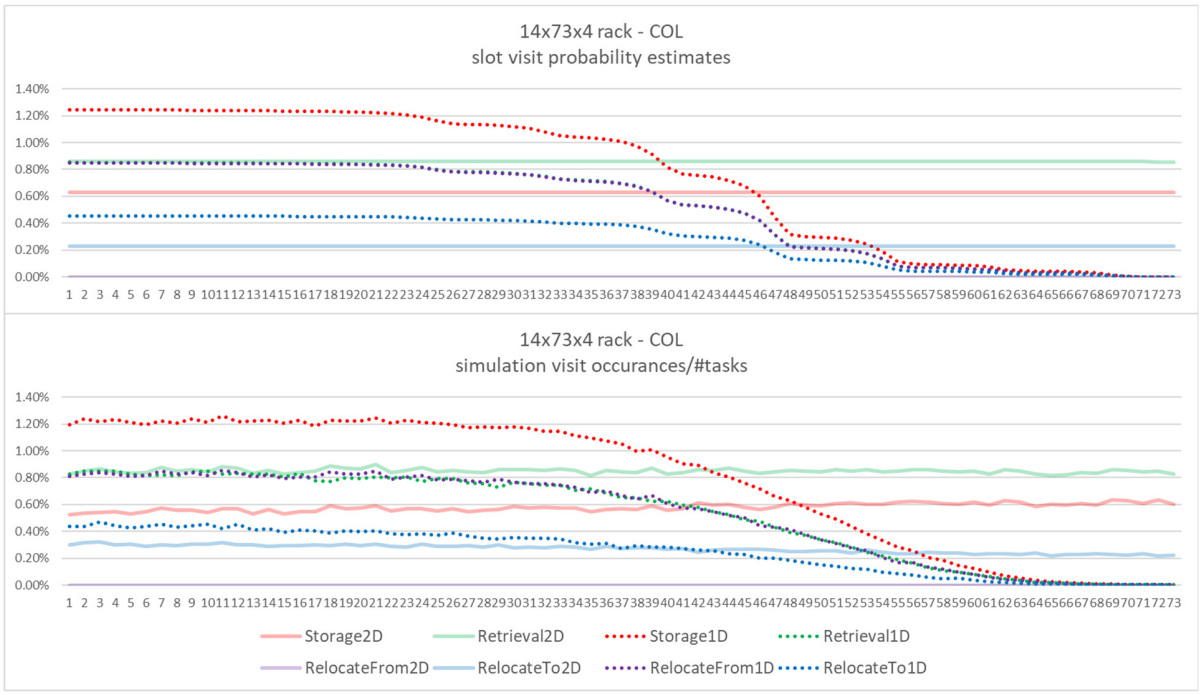

Figure 4.7 Task Visit Probability Estimates vs Simulation (COL) of 1-deep and 2-deep slots in a 14-tier, 73-column, 2-deep aisle

4.5 Service Time Modeling

In this section, we use queuing-network analysis to estimate the device utilizations (as indicators of the system's throughput performance) and the task cycle times (as indicators of the system's responsiveness performance). The slot task visiting probabilities and the relocation ratio θ^R obtained from the analysis described in the previous section are used as inputs for the service time computations. The main focus of this section is the system configurations with the most complicated mathematical forms in the scope of this research, where the system is tier-to-tier

configuration, the racks are 2-deep (thus tote relocation operations are considered), the tote lifts are 2-unit (thus tours with different sizes are considered), Closest Open Location (COL) policy is applied for storage assignment, and Dual Cycle (DC) policy is applied for shuttle scheduling. Simpler system configurations like tier-captive configurations, 1-deep racks, 1-unit tote lifts, pure-random storage policy and simple FCFS shuttle scheduling policy can all be adapted to the mathematical formulations purposed here.

In our analytical approaches, each SBS/RS aisle is viewed as multiple queuing systems configured in serial and in parallel. Both types of tote lifts are approximated as G/G/1 queuing systems, where the probabilities of 1-unit tours (non-full tours) and 2-unit tours (full-tours) are estimated, based on which the lift utilizations and task service/waiting times are estimated. In each aisle, the storage lift service's departure process is the arrival process to the shuttle services, and the shuttle services' departure process is the arrival process to the retrieval lift. The analysis of the shuttle service is more complicated – depending on the system's complexity, at most 12 shuttle service cases are identified. With tier-captive configurations, each shuttle is viewed as a G/G/1 queuing system, and the DC ratio θ^D is approximated to present the interleaving service pattern of the DC policy, based on which the shuttle utilizations and the service/waiting times of both storage tasks and retrieval tasks are estimated. With tier-to-tier configurations, the shuttles are transferred from tier to tier by the shuttle lift as needed. With each configuration, the shuttle service is approximated as a combination of a bounded M/M/c queuing system and multiple G/G/1 queuing systems. The tier-transfer ratio θ^T is approximated to determine the estimates for the portions of in-tier tasks and tier-to-tier tasks and their service/waiting times.

4.5.1 Tote Lift Service Analysis

In every aisle of the SBS/RS, a storage lift and a retrieval lift are responsible for the vertical inbound and outbound service of totes to/from the tiers. Both tote lifts interact with the encompassing system at the I/O floor. The two lifts interact with rack tiers at each tier's input buffer and output buffer, respectively. Multi-unit tote lifts are designed to transport multiple totes simultaneously in each tour. Such lifts are also called *multi-shuttle* lifts, while we choose to use the term *multi-unit* to avoid confusion with the horizontal shuttles. A *tour* is defined as the set of operations starting from the time when the idle lift starts moving from its current location to load the first tote, until the time when the lift completed unloading the last tote. Each tour is not

necessarily full, which means the lifts will not wait for totes that are not ready to be loaded. In this section the tote lifts under analysis are 2-unit, as the analysis of 1-unit lifts is relatively simple and can be inferred according to the analysis of 2-unit lifts, and lifts with capacity larger than two are not common for the SBS/RS systems. Operations of 2-unit storage lifts and retrieval lifts can both be summarized as in Figure 4.8, in which both the storage lift service and retrieval lift service are illustrated. The I/O floor is the load point from the storage lift's perspective, and the unload point from the retrieval lift's perspective, and the I/O buffers are the unload points from the storage lift's perspective, and load points from the retrieval lift's perspective.

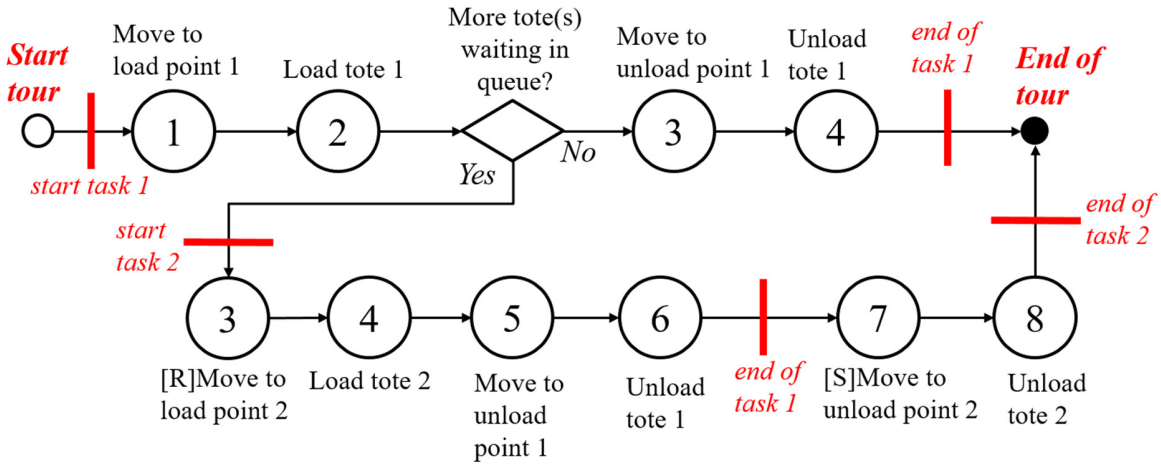

Figure 4.8 Tote Lift Service Flows (2-unit tote lift)

As introduced in Section 4.3the travel time between any two vertical tier locations $\tau^{TL}_{x1,\ x2}$ is assumed given, where $x1,\ x2 \in [0,\ X]$, and tier 0 indicates the I/O floor location. Load/unload times ω^{TL} are assumed identical for each tote regardless of its sequence in the tour. Travel time and L/U time parameters are assumed identical for both the storage lift and the retrieval lift of an aisle.

The multi-unit tote lifts can be approximated into G/G/1 queuing forms. It is obvious that the task arrival rates to both tote lifts are equal to the aisle's overall task arrival rate λ. The standard deviation of task inter-arrival time (IAT) for the storage lift σ_{SL} equals that of the overall storage tasks σ_S. However, the standard deviation of IAT for the retrieval lift σ_{RL} needs to be computed based on the queuing analysis results of the shuttle service (as illustrated in Figure 4.2). In this section generalized forms are used to present the analysis for both types of tote lifts.

The service time mean and variance for each task can be estimated based on the probabilities that the tote is in different cases with respect to the tote's tour size and its position in the tour. Denote p_x^{TL} as the probability that a task's *target tier* (thus the tier of a storage tote's target slot for the storage lift, or the current tier of a retrieval tote for the retrieval lift) is tier x. With slot task visit probabilities estimated in Section 4.4we have:

$$p_x^{TL} = \sum_y \sum_z P_{x,y,z}^S = \sum_y \sum_z P_{x,y,z}^R$$

The above equation can be presented as a simpler form $p_x^{TL} = 1/X$ under the earlier assumption that totes are randomly assigned to rack tiers with equal probabilities for both COL and PRS cases. Denote t_k and σ_k^2 as the mean and variance of task service time in a k-tote tour. For 2-unit tote lifts, the tour times are computed as following:

$$t_1 = \left(\sum_{x=1}^{L} \frac{2\tau_{0,\ x}^{TL}}{X} \right) + 2\omega^{TL}$$

$$\sigma_1^2 = \sum_{x=1}^{L} \frac{\left(t_1 - 2\tau_{0,\ x}^{TL} - 2\omega^{TL} \right)^2}{X}$$

$$t_2 = 0.5 \times \left(\left(\sum_{x1=1}^{L} \sum_{x2=1}^{L} \frac{2\tau_{0,\ x1}^{TL} + \tau_{x1,\ x2}^{TL}}{X^2} \right) + 4\omega^{TL} \right)$$

$$\sigma_2^2 = 0.5 \times \sum_{x1=1}^{L} \sum_{x2=1}^{L} \frac{(t_2 - 2\tau_{0,\ x1}^{TL} - \tau_{x1,\ x2}^{TL} - 4\omega^{TL})^2}{X^2}$$

Denote r_k as the probability that a task is served in a k-tote tour. Utilization of a K^{TL}-unit tote lifts can be presented as:

$$U = \lambda \sum_{k=1}^{K^{TL}} r_k t_k$$

Denote P_n as the probability that anytime there are n tasks in the queuing system of the tote lift, r_k can be estimated as:

$$r_k = \begin{cases} P_{k-1} & , \quad \forall k < K^{TL} \\ 1 - \sum_{k=1}^{K^{TL}-1} P_{k-1} & , \quad k = K^{TL} \end{cases}$$

For 2-unit tote lifts, when the variance of task inter-arrival time is significant, r_k can be approximated as:

$$r_1 = P_0 \approx (1 - U)$$

$$r_2 = 1 - P_0 \approx U$$

Because $U = \lambda(r_1 t_1 + r_2 t_2)$, r_k can be solved as:

$$r_1 = \frac{1 - \lambda t_2}{1 + \lambda(t_1 - t_2)}$$

$$r_2 = 1 - r_1 = \frac{\lambda t_1}{1 + \lambda(t_1 - t_2)}$$

Finally, the utilization U and average task service time T of a 2-unit tote lift can be obtained as:

$$U = \frac{\lambda t_1}{1 + \lambda(t_1 - t_2)}$$

$$T = r_1 t_1 + r_2 t_2 = \frac{t_1}{1 + \lambda(t_1 - t_2)}$$

Denote σ_a as the standard deviation of the overall task inter-arrival time. Denote σ_s as the standard deviation of the overall task service time, it is approximated as:

$${\sigma_s}^2 = r_1 {\sigma_1}^2 + r_2 {\sigma_2}^2 + r_1 (T - t_1)^2 + r_2 (T - t_2)^2$$

According to *Kingman's formula* for G/G/1 queuing system, the average task waiting time W is estimated as:

$$W \approx \frac{cov_a^2 + cov_s^2}{2} \times \frac{U^2}{1 - U} \times T, \qquad where\ cov_a = \frac{\sigma_a}{1/\lambda}, cov_s = \frac{\sigma_s}{T}$$

The mathematical formulations developed in this section also provide insights for the control strategy development phase. For example, the r_k computation indicates that when the task

arrival rate is low, a larger portion of "underloaded" tours is expected to occur, which is generally unfavorable because the device is less efficient in terms of average service time per task. Also, it is generally favorable to reduce the $\tau_{x1,\ x2}^{TL}$ elements in the service time equations in order to improve the overall service rate – for storage lifts, this can be obtained through advanced storage assignment policies; for retrieval lifts, this can be obtained through scheduling approaches.

Storage Lift

For the storage lift, because it is the first stage server of the storage tasks, there is $\sigma_a^{SL} = \sigma_a$ for the standard deviation of its task inter-arrival times. Thus, the queuing estimates U^{SL}, T^{SL}, σ_s^{SL} and W^{SL} are all obtained before the shuttle service analysis. At last, the standard deviation of storage tasks' inter-departure times from the storage lift is estimated in order to determine the inter-arrival time distribution to the shuttles in the next stage. It is computed based on the approximation solutions for Two-stage Tandem Queues by Rosenshine and Chandra (1975), in which the departure time variance σ_d^2 of the first stage of a tandem queue is approximated as:

$$\sigma_d^2 \approx 1/n\lambda^2 + (n-1)/n\mu^2 + (1-\rho)(n-1)/mn\mu^2 - (m-1)/m\mu^2$$
$$+ 0.5(1-\rho)(m-1)(n-1)/m^2n\mu^2 + 2(1-\rho)(m-1)(n-1)/mn^2\mu^2$$

In the above equation, λ, μ and ρ indicate the arrival rate, service rate, and utilization of the first stage server, $m = \frac{(\text{mean service time})^2}{\text{variance of service time}}$, $n = \frac{(\text{mean interarrival time})^2}{\text{variance of interarrival time}}$. It is noticeable that if both the inter-arrival time and service time are exponential (thus $m = n = 1$), the inter-departure times are also exponential and $\sigma_d = 1/\lambda$. In our content, replace σ_d^2 as $(\sigma_d^{SL})^2$ to denote the inter-departure time variance of the storage lift, thus λ, μ and ρ from the previous equation are replaced by λ, $\frac{1}{T^{SL}}$ and U^{SL}, respectively, and $m = \left(\frac{T^{SL}}{\sigma_s^{SL}}\right)^2$, $n = \left(\frac{1/\lambda}{\sigma^S}\right)^2$. Then σ_d^{SL} is computed accordingly. At last, denote $cov_d^{SL} = \lambda\sigma_d^{SL}$ as the coefficient of variation of the storage lift inter-departure times.

Retrieval Lift

For the retrieval lift, utilization U^{RL} and average service time T^{RL} can be directly estimated. However, because the standard deviation σ_a^{RL} of its inter-arrival time is estimated based

on the inter-departure time distribution of retrieval tasks σ_{dR}^{V} of the shuttles, the waiting time W^{RL} is obtained after the analysis for shuttle service (described in the following Section 4.5.2).

4.5.2 Shuttle and Shuttle Lift Service Analysis

Shuttles in SBS/RS serve both storage tasks and retrieval tasks. With 2-deep rack systems, a relocation operation is needed to retrieve a tote stored in a 2-deep slot if it is blocked by a tote stored in its neighboring 1-deep slot. With tier-to-tier systems, tier-transfer operations are needed to transport the shuttles between tiers by the shuttle lifts. The service flows of shuttle and shuttle lift can be summarized as in Figure 4.9.

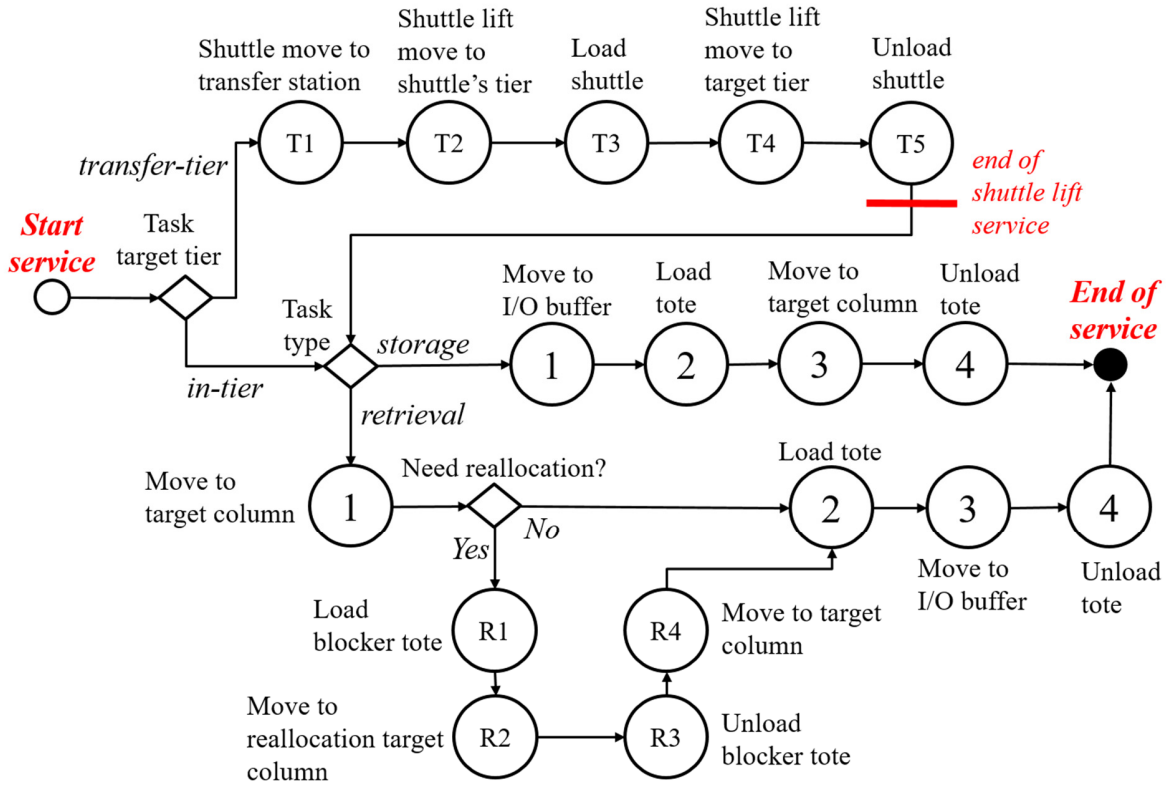

Figure 4.9 Shuttle and Shuttle Lift Service Flows

The shuttles' interaction points to tote lift services — input buffers and output buffers – are both located at the same end of each tier. In general, if the shuttles serve the tasks in a dual cycle (DC) interleaving pattern – storage, retrieval, storage, retrieval … – as much as possible, the shuttle service rate will be improved as the unnecessary travel times are minimized. In this research, an in-tier task is viewed as a DC task only if it matches one of the following conditions:

1. This task is a storage task and the previous task of the same shuttle going to serve it is a retrieval task (no matter in-tier or tier-transfer) in the same tier. In this case, the shuttle's current location is column 0 (I/O buffer), and there is no need for the shuttle to travel before it starts loading the tote of the current task.
2. This task is a retrieval task, and the previous task of the same shuttle going to serve it is a storage task (no matter in-tier or tier-transfer) in the same tier. In this case, the shuttle's current location is the column y_S of the last task's target slot $[x_S, y_S, z_S]$, and it needs to travel to the column y_R of the current task's target slot $[x_R, y_R, z_R]$ before it starts loading the tote of the current task.

Otherwise, an in-tier task is viewed as a single-cycle (SC) task. A tier-transfer task, either storage or retrieval, not counted as DC or SC. Relocation does not affect the DC/SC criterion for in-tier retrieval tasks.

According to definitions of task types, DC, tier-transfer and relocation, all possible shuttle services can be categorized into 12 cases based on the shuttle's service sequence, as illustrated in Table 4.4. Task types are denoted as *S* and *R* for storage and retrieval, respectively; *TT* and *Re* in the service sequences are used to indicate tier-transfer and relocation, respectively. Cases 1 to 4 are storage task cases, and cases 5 to 12 are retrieval task cases. For example, case 8 represents the situation that a shuttle whose previous task is retrieval (*R*) is selected to serve a tier-transfer (*TT*), relocation-needed (*Re*) retrieval task (*R*). The dual cycle column indicates whether the task in each case is viewed as a DC task or SC task or neither. Finally, the occurrence probability of each case is presented using the computational variables θ^T, θ^R and θ^D defined in Section 4.3. θ^R is estimated in Section 4.4, while θ^T and θ^D need to be estimated during the following queuing analysis.

Table 4.4 Shuttle Service Cases

Case	Task	Last Task	Service Sequence	Cycle Type	p_k Occurrence Probability / Task
1	*S*	*R*	*R-S*	DC	$(1-\theta^T)\theta^D$
2	*S*	*R*	*R-TT-S*	--	$\theta^T \times 0.5$
3	*S*	*S*	*S-S*	SC	$(1-\theta^T)(1-\theta^D)$

4	*S*	*S*	*S-TT-S*	--	$\theta^T \times 0.5$
5	*R*	*R*	*R-R*	SC	$(1-\theta^T)(1-\theta^R)(1-\theta^D)$
6	*R*	*R*	*R-Re-R*	SC	$(1-\theta^T)\,\theta^R(1-\theta^D)$
7	*R*	*R*	*R-TT-R*	--	$\theta^T(1-\theta^R) \times 0.5$
8	*R*	*R*	*R-TT-Re-R*	--	$\theta^T\theta^R \times 0.5$
9	*R*	*S*	*S-R*	DC	$(1-\theta^T)(1-\theta^R)\theta^D$
10	*R*	*S*	*S-Re-R*	DC	$(1-\theta^T)\,\theta^R\theta^D$
11	*R*	*S*	*S-TT-R*	--	$\theta^T(1-\theta^R) \times 0.5$
12	*R*	*S*	*S-TT-Re-R*	--	$\theta^T\theta^R \times 0.5$

Service Time Estimation by Case

The service procedures of the shuttle service cases are further illustrated in Figure 4.10 and Figure 4.11. For each service case, the distribution of shuttle service times can be estimated based on the task visit probability matrices P^S, P^R, P^{RelI} and P^{ReO} obtained from the previous stage. Under the earlier assumption that totes are randomly assigned to rack tiers with equal probabilities for both COL and PRS cases, the matrices are presented with y and z coordinates only. For example, $P^S_{y,z}$ indicates the storage probability to slot $[y,z]$ of any tier, thus $P^S_{y,z} = \sum_x P^S_{x,y,z}$. Furthermore, denote $P^S_y = \sum_x \sum_z P^S_{x,y,z}$ and $P^S_z = \sum_x \sum_y P^S_{x,y,z}$, etc. Denote t_k and σ_k^2 as the mean and variance of the service time of case k, where $k \in [1,2 \dots 12]$. The computation processes here are similar to those for tote lifts in Section 4.5.1. Hence, here we only use cases 1, 9, 10, 11 and 12 for illustration. In the following t_k equations, $[x0, y0, z0]$ and $[x1, y1, z1]$ indicates the target slots of the shuttle's previous task and current task, respectively, and $[x2, y2, z2]$ indicates the relocation target slot if applicable.

Case 1: $[\,R-S]$:

$$t_1 = \left(\sum_{y1=1}^{Y} \sum_{z1=1}^{Z} \left(P^S_{y1,z1} \times (\tau^V_{0,y1} + \omega^V_{z1}) \right) \right) + \omega^V_0$$

Case 9: $[\,S-R]$:

$$t_9 = \frac{\sum_{y0=1}^{Y}\sum_{y1=1}^{Y}\sum_{z1=1}^{2}(P_{y0}^{S} \times P_{y1,z1}^{R} \times (\tau_{y0,y1}^{V}+\tau_{y1,0}^{V} + \omega_{z1}^{V}))}{\sum_{z1=1}^{2} P_{z1}^{R}} + \omega_0^V$$

Note that case 1 and case 9 are the two basic DC cases. In both cases, the shuttle always interact with the I/O buffer once (L/U time ω_0^V) and interact with the slot once (L/U time ω_{z1}^V). In t_1 there is only one travel component $\tau_{0,y}^V$ for the travel time from the I/O buffer to the target column, this is because after unloaded a storage task, the shuttle does not need to perform additional travel before loading a retrieval task. On the other hand, t_9 has two travel elements: $\tau_{y0,y1}^V$ for the travel time from shuttle's current location to target column, and $\tau_{y1,0}^V$ for the travel time from target column to I/O buffer. The service time of the two basic SC cases: t_3 (S-S) and t_5 (R-R), can be inferred similarly.

Case 10: $[\,S - Re - R]$:

$$t_{10} = \left(\frac{\sum_{y0=1}^{Y}\sum_{y1=1}^{Y}\sum_{z1=3}^{4}(P_{y0}^{S} \times P_{y1,z1}^{R} \times (\tau_{y0,y1}^{V}+\tau_{y1,0}^{V} + \omega_{z1}^{V}))}{\sum_{z1=3}^{4} P_{z1}^{R}} + \omega_0^V\right)$$

$$+ \frac{\sum_{y1=1}^{Y}\sum_{z1=3}^{4}\sum_{y2=1}^{Y}\sum_{z2=1}^{4}(P_{y1,z1}^{ReO} \times P_{y2,z2}^{ReI} \times (2\tau_{y1,y2}^{V} + \omega_{z1}^{V} + \omega_{z2}^{V}))}{(\theta^R)^2}$$

Each relocation case has an additional relocation time component compared to the non-relocation cases. The above t_{10} equation describes the expected service time of a relocation-retrieval task given the shuttle's previous task is storage. Note that the first component of t_{10} is the same as t_9 except for the z indices. The second component of t_{10} is the time for the relocation operations, which consists of two travel elements ($2\tau_{y1,y2}^V$) and two L/U elements (ω_{z1}^V, ω_{z2}^V). The service time of cases t_6 (*R-Re-R*) can be inferred similarly by adding a relocation component based on t_5 (*R-R*).

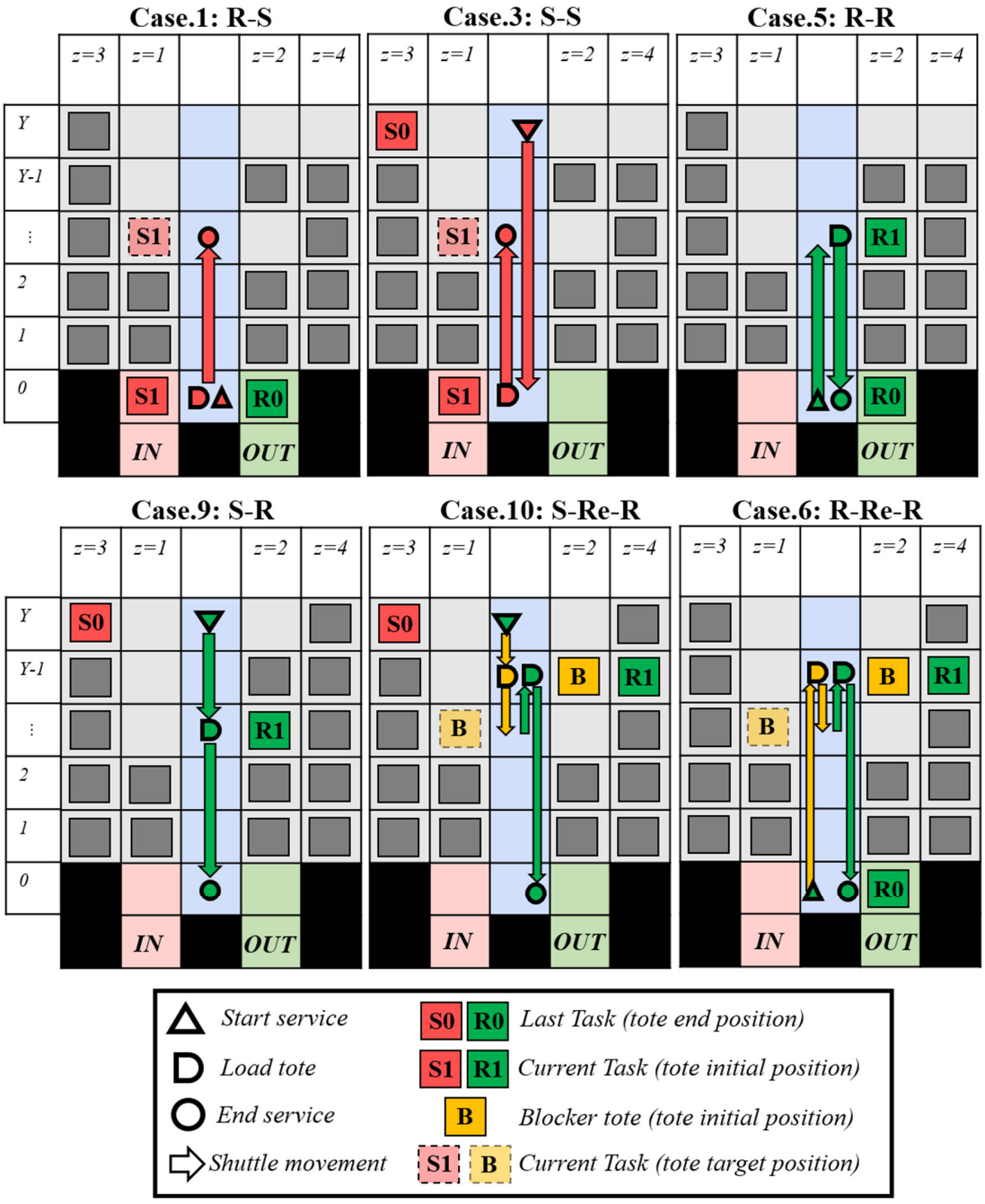

Figure 4.10 Shuttle service cases: in-tier tasks

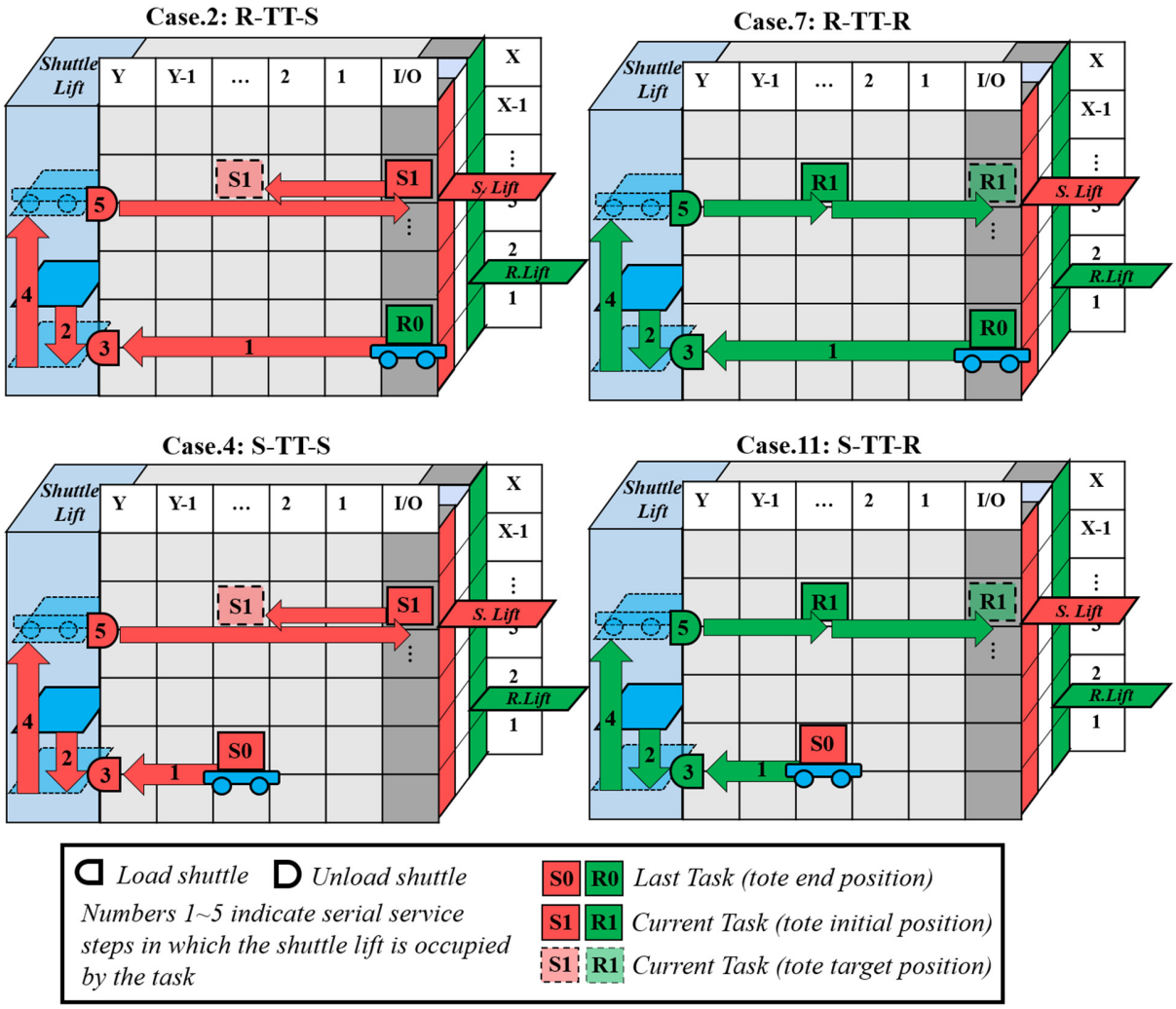

Figure 4.11 Shuttle service cases: tier-transfer tasks

Case 11: $[S-TT-R]$:

$$t_{11} = \left(\sum_{y0=1}^{Y}\left(P_{y0}^{S} \times \tau_{y0,Y}^{V}\right)\right) + \left(\frac{\sum_{x0=1}^{X}\sum_{x1=1}^{X} 2\tau_{x0,x1}^{VL}}{X^2} + 2\omega^{VL}\right) + \left(\left(\sum_{y1=1}^{Y}\sum_{z1=1}^{2}\left(P_{y1,z1}^{R} \times \left(\tau_{Y,y1}^{V} + \tau_{y1,0}^{V} + \omega_{z1}^{V}\right)\right)\right) + \omega_{0}^{V}\right)$$

For each tier-transfer task, the selected shuttle first travels to the transfer station located at column C (the opposite end of I/O buffers). As for t_{11}, the first component in the formula indicates the

average travel time from the column of the previous storage task to column Y (step 1 in Figure 4.11) – denote this component as t_{SY}^{V}. During this time, although the shuttle lift is idle, it is reserved by this task and waits until the selected shuttle arrived the transfer station (column Y). The second component indicates the total service time of the shuttle lift – denote this component as T^{VL} — consists of the processes where the shuttle lift travels toward the shuttle's current tier, loads the shuttle, travels toward the target tier of the task, and then unload the shuttle (step 2, 3, 4, 5 in Figure 4.11). According to the storage assignment assumption made previously that storage probabilities are equal between tiers, the two travel components are presented in the above form which consists of two travel elements ($2\tau_{x0,x1}^{VL}$) and two L/U elements ($2\omega^{VL}$). Also, according to the scheduling assumptions made previously, the shuttle lift will not start moving to the shuttle's tier until the shuttle is standing by at the transfer station – this is why the first two components in the formula are presented independently (otherwise the shuttle's travel time to the transfer station will partially overlap with the shuttle lift's travel time to the shuttle's current tier). The last component of the formula indicates shuttle retrieval operations after the shuttle is unloaded to the transfer station of the new tier, which consists of two travel elements ($\tau_{Y,y1}^{V}$, $\tau_{y1,0}^{V}$) and two L/U elements (ω_{z1}^{V}, ω_{0}^{V}). The t_4 (*S-TT-S*) can be inferred similarly by changing the last component in t_{11} above. The formula of t_7 (*R-TT-R*) is identical to t_{11} above except for the first component for the travel time from the shuttle's position after completed the previous retrieval task to column Y – denote this component as t_{RY}^{V}, which simply equals $\tau_{0,Y}^{V}$ because in this case the selected shuttle will always need to travel along the full aisle from I/O buffer to the transfer station. The t_2 (*R-TT-S*) can be inferred similarly by changing the last component of t_7.

For the rest of the cases, both tier-transfer and relocation exist in the service. Their service times can all be determined by adding a relocation component to the corresponding basic tier-transfer case. For example, case #12 [$S - TT - Re - R$], which is the most complicated case among all 12 shuttle service cases, is represented:

Case 12: [$S - TT - Re - R$]:

$$t_{12} = \left(\sum_{y0=1}^{Y} \left(P_{y0}^{S} \times \tau_{y0,Y}^{V} \right) \right) + \left(\frac{\sum_{x0=1}^{X} \sum_{x1=1}^{X} 2\tau_{x0,x1}^{VL}}{X^2} + 2\omega^{VL} \right)$$

$$+\frac{\sum_{y1=1}^{Y}\sum_{z1=3}^{4}\sum_{y2=1}^{Y}\sum_{z2=1}^{4}(P_{y1,z1}^{ReO}\times P_{y2,z2}^{ReI}\times\left(2\tau_{y1,y2}^{V}+\omega_{z1}^{V}+\omega_{z2}^{V}\right))}{(\theta^{R})^{2}}$$

$$+\left(\left(\sum_{y1=1}^{Y}\sum_{z1=1}^{2}\left(P_{y1,z1}^{R}\times\left(\tau_{Y,y1}^{V}+\tau_{y1,0}^{V}+\omega_{z1}^{V}\right)\right)\right)+\omega_{0}^{V}\right)$$

As showed above, t_{12} has four components: shuttle travels from previous storage location to transfer station (which is identical to the first component of t_{11}), shuttle lift service time (identical to the second component of t_{11}), relocation time (identical to the second part of t_{10}), and retrieval time (identical to the third part of t_{11}).

Some important observations are made regarding tier-transfer tasks and the ratio θ^{T}. It is reasonable to view the tier-transfer services as a virtual independent queuing system – call it the Tier-transfer system (TT system) – whose task arrival rate is $2\lambda\theta^{T}$. Denote the utilization of the TT system as U^{TT}, which can be presented as:

$$U^{TT}=2\lambda\theta^{T}(T^{VL}+0.5(t_{SY}^{V}+t_{RY}^{V})),$$

Where t_{SY}^{V} and t_{RY}^{V} are the average times that a shuttle travels from its current position to the tier-transfer station after a storage task and a retrieval task, respectively, and T^{VL} is the average service time of the shuttle lift – travel to shuttle's tier, load shuttle, travel to target tier, and unload shuttle. Note that in the above presentation for U^{TT}, there is only one unknown variable θ^{T} (as discussed in the cases' service time computations, T^{VL}, t_{SY}^{V} and t_{RY}^{V} are obtained without the other unknown variable θ^{D}, and θ^{R} is already known). According to the control assumptions made previously, a task whose target tier does not have shuttle availability will initiate a tier-transfer service only when there is at least one idle shuttle on the other tiers. It can be inferred that θ^{T} will be in general smaller as λ gets higher (which causes higher shuttle utilization), and vise-versa. The negative correlation between λ and θ^{T} raises complexity in analyzing this virtual system. To some extent, the TT system behaves similar to a J-sever queuing system, where J is the number of shuttles. However, once a tier-transfer service starts, the shuttle lift is also occupied together with the selected shuttle until the shuttle is released to the target tier – during this time, no other tier-transfer tasks can be started in parallel with the current task even if there are more idle shuttles (but these shuttle will still go for other in-tier tasks if any arrives during this time), which add some single-

server queuing system characteristics to it. Moreover, the queue capacity of the TT system is limited to $(X-J)$ in an X-tier, J-shuttle aisle, because obviously each tier without shuttle availability can only have at most one tier-transfer task at any time (if there are multiple tasks waiting on that tier, only the first task will initiate tier-transfer service). The analysis of the TT system and the approximation of θ^T will be introduced later in this section.

At last, the variance of service time is computed for each of the 12 cases. In general, these are computed as:

$$\sigma_k^2 = \sum_{i(k)} \left(p_{i(k)} \times \left(t_k - t_{i(k)} \right)^2 \right)$$

Where k is the case index, $i(k)$ represents the ith member within case k, and $t_{i(k)}$ and $p_{i(k)}$ indicate its service time and probability within case k, respectively. For example, in case 9 (S-R) there are:

$$t_{i(9)} = \left(\tau^V_{y0,y1} + \tau^V_{y1,0} + \omega^V_{z1} \right) + \omega^V_0$$

$$p_{i(9)} = \frac{P^S_{y0} \times P^R_{y1,z1}}{\sum_{z=1}^{2} P^R_{y1,z1}}$$

$$y0 \in [1, Y], y1 \in [1, Y], z1 \in [1,2]$$

The computation details here are similar for all shuttle service cases, hence we do not discuss them in detail. Finally, although the mean and variance computation processes may look cumbersome at first glance, they are actually easy for coding implementation and are computational efficient in the analysis program as introduced in section 4.3.3 (anyway, the travel-time model developed here is not expected for manual computation).

Estimation of Ratios θ^D and θ^T

Recall that instead of specifying whether a dual cycle should start with a storage or retrieval task, we define that a storage or retrieval task is considered as a DC task if the previous task served by the same shuttle is of the different type, and considered as an SC task otherwise. The mean and variance of each shuttle service case are estimated without the dual-cycle ratio θ^D or the tier-transfer ratio θ^T. However, to compute for the overall service time, these two ratios need to be

estimated in order to know the probabilities of each case. Based on the shuttle scheduling assumption made in Section 4.3 if the shuttles serve tasks in simple FCFS pattern ($M_{DC} = false$), there is expected to be equal number of DC tasks and SC tasks, thus θ^D is 0.5. However, with DC scheduling, θ^D is affected by shuttle utilization – in general, the busier the shuttles are, the higher θ^D is. The lower bound of θ^D with DC scheduling is 0.5, which is obvious. On the other hand, θ^T is affected by the number of tiers and number of shuttles deployed, the shuttle utilization, and the shuttle lift utilization as well. The lower bound of θ^T is 0.

A nested goal-seeking approach is developed to approximate both θ^D and θ^T. Define p_k as the probability of occurrence for each service case $k \in [1,12]$. Each p_k is a function of θ^D and θ^T as described in Table 4.4 (relocation ratio θ^R is already known). The goal-seeking approach is illustrated in Figure 4.12. For tier-to-tier systems with DC scheduling mode, both of the ratios need to be estimated. The initial values of θ^T and θ^D are set to 0 and 1, respectively: $\theta^T = 0$ means no tier-transfer task will occur, and $\theta^D = 1$ means the shuttles are always able to serve all tasks in DC patterns. The goal-seeking approach is essentially the processes of increasing the input θ^T and decreasing the input θ^D until both values are consistent with their corresponding estimates obtained in iterative computations. Estimates $\widehat{\theta^T}$ and $\widehat{\theta^D}$ are computed and compared to the current θ^D and θ^T values, respectively. An estimate is accepted if the error is within preset allowances, otherwise the estimates are rejected and the θ value is updated for a new round of estimation – though such iterative approximation processes, the final values of θ^D and θ^T are determined. In this research, the acceptance allowances of θ^D and θ^T are determined as 0.25% and 0.5%, respectively. The maximum increment/decrement of each update are determined as 0.1% and 0.2% for θ^D and θ^T, respectively. The details of the two estimation functions are described in the following pseudo-code forms.

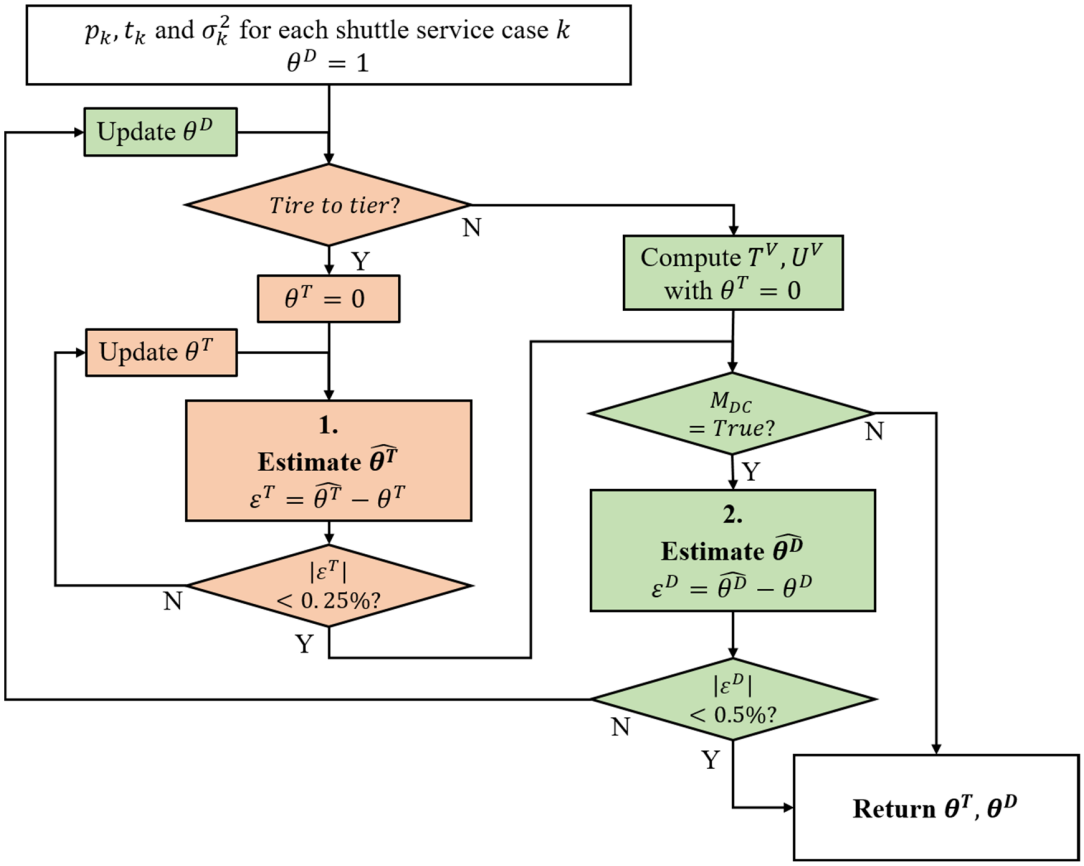

Figure 4.12 Goal-seeking for DC ratio θ^D and tier-transfer ratio θ^T

Goal-seeking function 1: Estimate for $\boldsymbol{\theta^T}$:

$$P^{noShuttle} = (X - J)/X$$

$$UB(\theta^T) = \min\left[\frac{1}{2\lambda\left(T^{VL} + 0.5(t_{SY}^V + t_{RY}^V)\right)}, P^{noShuttle}\right]$$

$$U^{VL} = 2\lambda\theta^T \times T^{VL}$$

$$T^V = 0.5\sum_{k=1}^{12}(p_k \times t_k)$$

$$U^V = \frac{2\lambda}{J} \times T^V$$

$$P(0) = 1\Big/\left[\sum_{m=0}^{J-1}\frac{(J \times U^V)^m}{m!} + \frac{(J \times U^V)^J}{J! \times (1 - U^V)}\right]$$

FOR $(n = 1,2, \ldots, X - J)$:

$$P(n) = \begin{cases} P(0) \times \dfrac{(J \times U^V)^n}{n!}, & if\ n < J \\ P(0) \times \dfrac{(J \times U^V)^n}{J^{(n-J)} \times J!}, & otherwise \end{cases}$$

END FOR

$P^{idleVL} = 1 - U^{VL}$

$$P^{busyVL\_diffTier} = U^{VL} \times \sum_{n=1}^{X-J} \left(P(n) \times \left(1 - \frac{n}{X-J}\right)\right)$$

$\widehat{\theta^T} = min[P^{noShuttle} \times (P^{idleVL} + P^{busyVL\_diffTier}),\ UB(\theta^T)]$

$\varepsilon^T = \widehat{\theta^T} - \theta^T$

IF ($|\varepsilon^T| < 0.0025$):
    STOP
ELSE IF ($\varepsilon^T < 0$):
    Update $\theta^T = \theta^T + min(0.1\varepsilon^T, -0.001)$
ELSE:
    Update $\theta^T = \theta^T + max(0.1\varepsilon^T, 0.001)$
END IF

In the above function for θ^T estimation, the upper bound of θ^T is estimated at the first place. Denote $P^{noShuttle} = (X - J)/X$ for the probability that a task is targeting a tier that has no shuttle existence when it arrives, where *J* is the number of shuttles and *X* is the number of tiers. It can be inferred that, there are two cases that such task will become a tier-transfer task:

1) The shuttle lift is currently idle, thus this task will be the next task the shuttle lift serves;

2) Although the shuttle lift is currently busy, all existing tier-transfer tasks waiting for shuttle lift service are targeting at tiers that are different from this task's target tier. Otherwise, this task will be an in-tier task because a shuttle will be transferred to its target tier in one of the previous tasks.

It is obvious that $\theta^T < (X - J)/X$. Also, θ^T is bounded by the maximum service rate of the virtual TT system which can be estimated given its utilization, $U^{TT} = 1$ – in such case, the shuttle lift is always occupied by one tier-transfer task (either busy or waiting for an incoming shuttle). The upper bound $UB(\theta^T)$ is thus obtained. From the perspective of the TT system, $U^{TT} = 1$ does not lead to infinite queues: it can be inferred that at any moment in the aisle, the target tiers of all existing tier-transfer tasks are different, thus the maximum possible number of tier-transfer tasks in the aisle is limited to $(X - J)$.

Then, based on the current value of θ^T, shuttle lift utilization U^{VL} is estimated according to the average service time T^{VL} estimated earlier and the arrival rate of tier-transfer tasks based on the current value of θ^T. The shuttle utilization U^V and expected overall service time T^V are estimated based on the current values of θ^T and θ^D (because case probabilities p_k are functions of both ratios). $P(n)$'s are the probabilities of number of tier-transfer tasks in the entire system, estimated by viewing the overall tier-transfer service process as an M/M/c ($c = J$) queuing system (note that the tier-transfer service was temporarily viewed as single-server only for obtaining the upper bound $UB(\theta^T)$). Then, for each tier-transfer task arrival, two probability components are formed to estimate the probability that the arriving task initializes a tier-transfer service. Denote P^{idleVL} as the probability that the shuttle lift is idle, which is simply $1 - U^{VL}$. Denote $P^{busyVL\_diffTier}$ as the probability that the shuttle lift is busy while the target tier of the arriving task is different from that of any existing tier-transfer tasks in the system, estimated as U^{VL} multiplied by the weighted sum of the corresponding probabilities of all possible number-in-system cases. Take a 12-tier and 6-shuttle aisle for an example, obviously there will be 6 tiers not having shuttle all the time. If there are three tier-transfer tasks in the system (probability $P(3)$) by the time a task arrives(known its target tier does not have shuttle now), then 3 out of the 6 no-shuttle tiers are expecting shuttles transfers prior to this task – because the existing 3 tier-transfer tasks are targeting at the other three no-shuttle tiers, respectively, as inferred earlier. Thus, the probability that this task will become a tier-transfer task is expected to be $1 - 3/(12 - 6) = 0.5$.

Finally, an estimate $\widehat{\theta^T} = P^{noShuttle} \times (P^{idleVL} + P^{busyVL\_difftier})$ is computed and compared to the upper bound $UB(\theta^T)$ – if $\widehat{\theta^T}$ is larger, its value is set to $UB(\theta^T)$. The final $\widehat{\theta^T}$ is compared to the current θ^T. According to the error ε^T between $\widehat{\theta^T}$ and the current θ^T, the function either accept the current θ^T value or update it for a new iteration.

Goal-seeking function 2: Estimate for $\boldsymbol{\theta^D}$:

$$P^{idle\_DC} = 0.5$$

$$P^{storage\_insystem} = P^{retrieval\_insystem} = U^V$$

$$P^{storage\_inqueue} = P^{retrieval\_inqueue} = (U^V)^2$$

$$P^{busy\_currentIsIntier\_SC} = P^{storage\_insystem} \times (1 - P^{retrieval\_inqueue})$$

$$= P^{retrieval\_insystem} \times (1 - P^{storage\_inqueue})$$

$= U^V \times (1 - (U^V)^2)$

$P^{busy\_currentIsIntier\_DC} = 1 - P^{busy\_currentIsIntier\_SC} = (U^V)^3 - U^V + 1$

IF (Tier-Captive):

$P^{busy\_DC} = P^{busy\_currentIsIntier\_DC}$

ELSE IF (Tier-to-Tier):

$P^{busy\_DC} = (1 - \theta^T) \times P^{busy\_currentIsIntier\_DC} + \theta^T \times 0.5$

END IF

$\widehat{\theta^D} = P^{idle\_DC} \times (1 - U^V) + P^{busy\_DC} \times U^V$

$\varepsilon^D = \widehat{\theta^D} - \theta^D$

IF ($|\varepsilon^D| < 0.005$):
STOP
ELSE IF ($\varepsilon^T < 0$):
Update $\theta^D = \theta^D + min(0.2\varepsilon^D, -0.002)$
ELSE:
Update $\theta^D = \theta^D + max(0.2\varepsilon^D, 0.002)$
END IF

In the above function for θ^D estimation, service processes of the shuttles are temporarily viewed as independent M/M/1 queuing systems. The probabilities that a task is in the shuttle service system, and the probabilities that a task in the shuttle service queue, for both storage tasks and retrieval tasks. When the shuttle is idle, an arriving task is DC if the last task served by this shuttle is the different task type — this probability P^{idle_DC} is simply 0.5 because the type of the arriving task is independent from the type of the previous task. For tier-captive configurations where there is no tier-transfer task, when the shuttle is busy, the probability that the arriving task is SC P^{busy_SC} is approximated as the probability there is at least one task of the same type in the system multiplied by the probability that there is no task of the different type in the queue—for example, an arriving storage task will be SC if the shuttle is busy working on another storage task while there is no retrieval tasks in queue. For tier-to-tier configurations, the computations for P^{busy_SC} incorporated the concerns on whether the current task the shuttle is working on is in-tier or tier-transfer – when the shuttle is busy working on a tier-transfer task, the type of the arriving task is expected to be independent from the type of the current tier-transfer task (according to the scheduling control assumptions). Then, for both tier-captive cases and tier-to-tier cases, it is obvious that the probability that an arriving task to a busy shuttle is DC P^{busy_DC} equals $1 - P^{busy\_SC}$. Finally, an estimate $\widehat{\theta^D} = P^{idle\_DC} \times (1 - U^V) + P^{busy\_DC} \times U^V$ is

computed and compared to the current θ^D. According to the error ε^D between $\widehat{\theta^D}$ and the current θ^D, the function either accept the current θ^D value, or update it and start over a new iteration (note that θ^T is also initialized).

Inter-arrival Time Distributions

It is obvious that for each shuttle, the mean inter-arrival times are J/λ for both task types, thus the overall mean inter-arrival time is $J/2\lambda$. The standard deviation of each shuttle's task inter-arrival times are denoted as σ_{aS}^V and σ_{aR}^V for storage tasks and retrieval tasks, respectively. Based on the storage assignment assumptions and shuttle scheduling assumptions made previously, the probability that each storage task is assigned to any particular shuttle is equally $1/J$ (for both tier-captive cases and tier-to-tier cases). Denote task s_0 as the latest storage task assigned to shuttle j. Denote tasks $s_1, s_2 \ldots s_i$ as storage tasks departs from the storage lift after s_0, and denote $t_1, t_2 \ldots t_i$ as their departure times (relative to the departure time of s_0). Thus, the inter-arrival time of the next storage task assigned to shuttle j after task s_0 is t_{i*} of the earliest task s_{i*}which will also be served by shuttle j. Denote the probability $i = i^*$ as p_i. Because each departure task s_0 has equal probability $1/J$ assigned to each shuttle, there is:

$$p_i = \frac{1}{J}(1 - \frac{1}{J})^{i-1}, \forall i$$

Let us temporarily assume $\sigma_d^{SL} = 0$ for the purpose of computing the lower bound of the standard deviation σ_{aS}^V of storage arrivals for shuttle service, thus $t_i = i/\lambda$. It is obvious that the average storage inter-arrival time to each shuttle is J/λ. The lower bound of standard deviation σ_{aS}^V is thus estimated as:

$$LB({\sigma_{aS}^V}^2) = \sum_{i=1}^{\infty}\left[p_i \times \left(t_i - \overline{t_i}\right)^2\right] = \sum_{i=1}^{\infty}\left[\frac{1}{J}\left(1-\frac{1}{J}\right)^{i-1} \times (i/\lambda - J/\lambda)^2\right]$$

$$= \frac{1}{J\lambda^2} \times \sum_{i=1}^{\infty}\left[\left(1-\frac{1}{J}\right)^{i-1} \times (i-J)^2\right]$$

The latter part of the above formulation is a convergent *Taylor series*, which can be derived into the following form:

$$LB(\sigma_{aS}^{V\,2}) = \frac{1}{J\lambda^2} \times (J^2 \times (J-1)) = \frac{J(J-1)}{\lambda^2}$$

And thus the lower bound of the standard deviation is obtained as followed:

$$LB(\sigma_{aS}^{V}) = \sqrt{LB(\sigma_{aS}^{V\,2})} = \frac{\sqrt{J(J-1)}}{\lambda}$$

The lower bound of the coefficient of variation of storage inter-arrival times to each shuttle is estimated as followed. It can be inferred that the more shuttles in the aisle, the higher the coefficient of variation of the storage inter-arrival times, and cov_{aS}^{V} approximate 1 when number of shuttles J becomes relatively large – in such cases, the storage inter-arrival times to each shuttle is approximately exponential.

$$LB(cov_{aS}^{V}) = \frac{LB(\sigma_{aS}^{V})}{J/\lambda} = \sqrt{\frac{J-1}{J}}$$

The lower bound $LB(cov_{aS}^{V})$ is computed based on the assumption that $\sigma_d^{SL} = 0$. In this research, according to experiment results based on Monte-Carlo approaches, we assume that the following equations are a reasonable approximations of cov_{aS}^{V} and σ_{aS}^{V}:

$$cov_{aS}^{V} \approx max(LB(cov_{aS}^{V}) \ , \ cov_d^{SL})$$

$$\sigma_{aS}^{V} \approx \frac{J}{\lambda} \times cov_{aS}^{V}$$

This lower bound computation approach also applies for the retrieval task arrivals, because the probability that each retrieval task is assigned to any particular shuttle is also equally $1/J$. It can be inferred that for each shuttle, the lower bound of the coefficient of variation of retrieval inter-arrival times is the same as that of storage inter-arrival times, thus:

$$LB(cov_{aR}^{V}) = \sqrt{\frac{J-1}{J}}$$

Similarly, we assume that the following equations are a reasonable approximations of cov_{aR}^{V} and σ_{aR}^{V}:

$$cov_{aR}^{V} \approx max(LB(cov_{aR}^{V}) \ , \ cov_{R}), where \ cov_{R} = \lambda\sigma_{R}$$

$$\sigma_{aR}^{V} \approx \frac{J}{\lambda} \times cov_{aR}^{V}$$

Finally, the overall standard deviation of task inter-arrival times σ_a^V to each shuttle is estimated as:

$$\sigma_a^V = \sqrt{0.5{\sigma_{aS}^V}^2 + 0.5{\sigma_{aR}^V}^2}$$

Service Time Distributions

In the previous part of this section, mean service time T^V and utilization U^V are both obtained for shuttle service. Also, the mean t_k, variance σ_k^2 and occurrence probability p_k are obtained for each shuttle service case $k \in [1,12]$. The overall standard deviation of shuttle service times σ_s^V is thus estimated as:

$$\sigma_s^V = \sum_{k=1}^{12} [\, p_k\sigma_k^2 + p_k(T^V - t_k)^2 \,]$$

Note that in the above equation, both the variance components within each case and the variance components to the overall averages are incorporated. We decide to use such approach to approximate the variances of the shuttle service pattern described previously (Figure 4.9). Then, denote T_S^V and T_R^V as the mean shuttle service times for storage tasks and retrieval tasks, respectively, where:

$$T_S^V = \sum_{k=1}^{4} (p_k \times t_k)\,, \qquad T_R^V = \sum_{k=5}^{12} (p_k \times t_k),$$

$$T^V = 0.5 \times \sum_{k=1}^{12} (p_k \times t_k) = 0.5(T_S^V + T_R^V)$$

The standard deviation of each shuttle's service times are denoted as σ_{sS}^V and σ_{sR}^V for storage tasks and retrieval tasks, respectively, computed as:

$$(\sigma_{sS}^V)^2 = \sum_{k=1}^{4} [\, p_k \sigma_k^2 + p_k (T_S^V - t_k)^2 \,], \qquad (\sigma_{sR}^V)^2 = \sum_{k=5}^{12} [\, p_k \sigma_k^2 + p_k (T_R^V - t_k)^2 \,]$$

The corresponding coefficients of variations of shuttle service times are then estimated as:

$$cov_{sS}^V = \sigma_{sS}^V / T_S^V, \qquad cov_{sR}^V = \sigma_{sR}^V / T_R^V$$

Queuing Time Estimation

As the overall inter-arrival time to each shuttle has mean $J/2\lambda$, the shuttle utilization $U^V = 2\lambda T^V / J$. When DC policy is applied for shuttle scheduling, because the shuttles serve the tasks in interleaving patterns, the queue lengths of storage tasks and retrieval tasks waiting for shuttle services need to be estimated separately. Denote Q_S^V and Q_R^V as the estimates of the queuing lengths of a single shuttle. By viewing each shuttle as an independent G/G/1 queuing system, we purpose the following approach in approximating Q_S^V and Q_R^V according to According to *Kingman's formula*:

$$Q_S^V \approx \frac{{cov_{aS}^V}^2 + {cov_{sS}^V}^2}{2} \times \frac{(U^V)^2}{1 - U^V} \times \frac{1}{2}$$

$$Q_R^V \approx \frac{{cov_{aR}^V}^2 + {cov_{sR}^V}^2}{2} \times \frac{(U^V)^2}{1 - U^V} \times \frac{1}{2}$$

The above equations provide approximates of the queues by temporarily viewing each shuttle as two servers: one serves storage tasks and another serves retrieval tasks, while both servers share the same utilization U^V. Note that we make this assumption that this approximation approach is acceptable based on observations from the Monte-Carlo simulation results rather than based on mathematical proofs. Denote Q_{Sx}^V and Q_{Rx}^V as the expected number of in-tier tasks waiting for shuttle services on each tier, and denote Q_{TT}^V as the expected number of tier-to-tier tasks in the entire aisle (either storage or retrieval). For tier-captive configurations, there are obviously $Q_{Sx}^V = Q_S^V$ and $Q_{Rx}^V = Q_R^V$, and $Q_{TT}^V = 0$. However, for tier-to-tier configurations, Q_S^V and Q_R^V are the collective estimates for waiting tasks of a particular shuttle including the following three situations (illustrated in Figure 4.13):

1. Waiting in-tier tasks on the shuttle's current tier – the DC policy applies to those tasks;

2. Waiting tier-transfer tasks which will initiate tier-transfer operations with the shuttle – as introduced earlier, those tasks are only assigned to idle shuttles, and the DC policy does not apply to those tasks;
3. Waiting in-tier tasks on other tiers which will be served by a shuttle after the shuttle has completed a tier-transfer task. Although these tasks are not on the shuttle's current tier, they are viewed as in-tier tasks because they do not initiate tier-transfer operations. After the shuttle arrived their tiers, the DC policy also applies to those tasks.

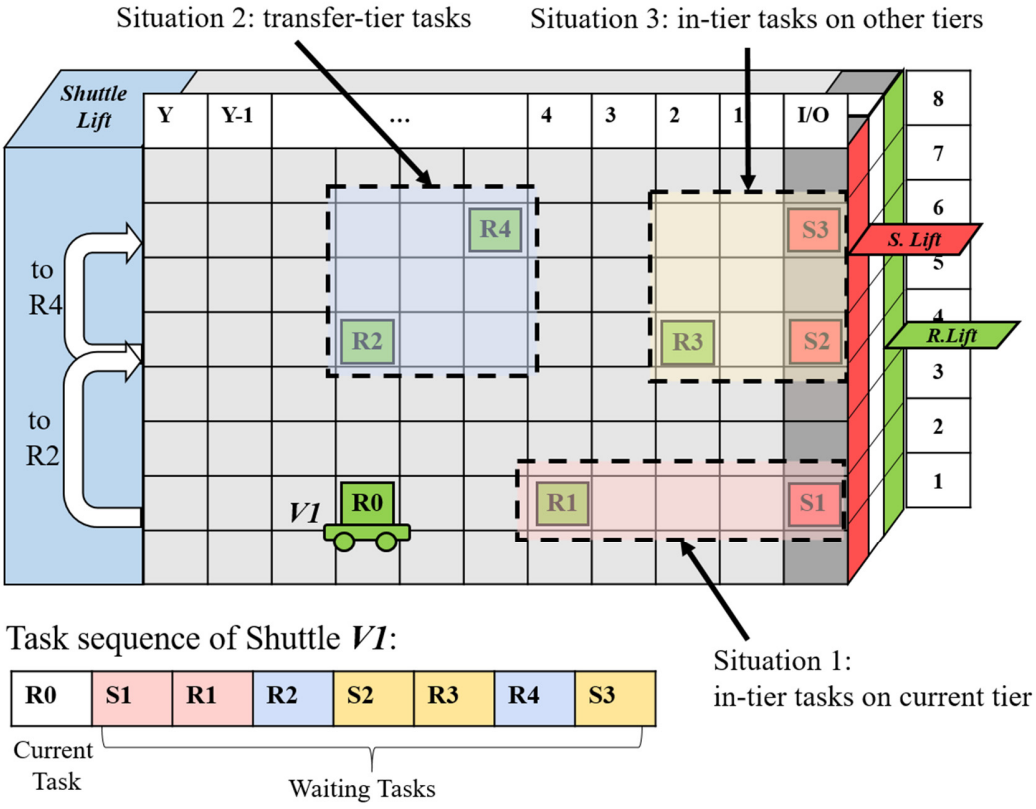

Figure 4.13 Different situations of waiting queues for shuttle services

At a particular moment in the system, shuttle V1 is currently serving task R0, and its future task service sequence is as illustrated. The queue estimates Q_S^V and Q_R^V include the tasks in all three situations. Known the tier-transfer ratio θ^T, the queues of total in-tier tasks of each shuttle (thus the sum of queues in situation 1 and 3) are estimated as $(1-\theta^T)Q_S^V$ and $(1-\theta^T)Q_R^V$ for storage and retrieval tasks, respectively. Because in steady-state, the queues of waiting tasks on each tier $x \in [1,2\ldots,X]$ are expected to be the same over tiers (for both task types), it can be inferred that:

$$Q_{Sx}^{V} = (1 - \theta^{T})Q_{S}^{V} \times \frac{J}{X}$$

$$Q_{Rx}^{V} = (1 - \theta^{T})Q_{R}^{V} \times \frac{J}{X}$$

Where J is the number of shuttles deployed and X is the number of tiers. Because DC policy only applies to in-tier tasks (situation 1 and 3), the queues on the shuttle's current tier Q_{Sx}^{V} and Q_{Rx}^{V} are considered as the only factors for estimating the task waiting times. Obviously, there are $Q_{Sx}^{V} = Q_{S}^{V}$ and $Q_{Rx}^{V} = Q_{R}^{V}$ in tier-captive configurations. In tier-to-tier systems, tier-transfer tasks are only assigned to idle shuttles and not considered in DC scheduling (situation 2). As introduced previously, tier-transfer service is viewed as a virtual TT system which resembles a bounded M/M/c queuing system, thus the queue length and waiting times of tier-transfer tasks are also estimated differently from those for the in-tier tasks. Consistent with the approaches applied in approximating for θ^{T}, the expected queue length Q_{TT}^{V} and the expected waiting time W_{TT}^{V} of tier-transfer tasks waiting for shuttle and shuttle lift service are estimated as follows:

$$Q_{TT}^{V} = \left[\sum_{n=1}^{X-J+1} nP(n)\right] + (X - J)\left(1 - \sum_{n=0}^{X-J+1} P(n)\right) - U^{TT}$$

$$W_{TT}^{V} = \frac{Q_{TT}^{V} \times U^{TT}}{2\lambda\theta^{T}}$$

In which U^{TT} is the utilization of the virtual Tier-transfer System (as introduced previously), where $U^{TT} = 2\lambda\theta^{T}(T^{VL} + 0.5(t_{SY}^{V} + t_{RY}^{V}))$. Essentially, Q_{TT}^{V} is estimated as the difference of the expected number of tasks in system and the expected number of tasks being served. The number-in-system estimation is based on approximating the virtual TT system as M/M/c ($c = J$) when a tier-transfer task is waiting for shuttles' availabilities. $P(n)$'s are the probabilities of number of tasks in the TT system, estimated as followed:

$$P(0) = 1 \Big/ \left[\sum_{m=0}^{J-1} \frac{(J \times U^{V})^{m}}{m!} + \frac{(J \times U^{V})^{J}}{J! \times (1 - U^{V})}\right]$$

$$P(n) = \begin{cases} P(0) \times \dfrac{(J \times U^V)^n}{n!}, & if\ 0 < n < J \\ P(0) \times \dfrac{(J \times U^V)^n}{J^{(n-J)} \times J!}, & otherwise \end{cases}$$

The expected number in system is estimated as the weighted sum of number in system by the corresponding probabilities. Because the maximum number in the TT system is limited to $(X - J)$ by definition of tier-transfer task, probabilities with $n > (X - J)$ are accumulated to $n = (X - J)$ in order to approximate the patterns of the tier-transfer services. Then, once a shuttle is selected, the system no longer allow other tier-transfer tasks to be served in parallel – thus the expected number of tasks in service equals U^{TT}. The expected queue length of overall tier-transfer tasks in the aisle Q_{TT}^V is thus estimated, and the waiting time W_{TT}^V is then estimated based on *Little's Law*.

The waiting time estimation is complex for dual cycle mode, as each shuttle serves its task queues in interleaving patterns. In Figure 4.14, the waiting time situations are illustrated for an arrival task n^* depending on the type of task n^*, the type and progress of the shuttle's current service task (s_0 or r_0), the shuttle's current queues of both task types, and also the future task arrivals after task n^*. In the examples illustrated in this figure, the retrieval task queue is larger (three) than the storage task queue (two) at the moment that task n^* arrives. Because the DC policy only applies when the shuttle is busy, the expected lengths of the two queues are estimated as $Q_{Sx}^V/\mathrm{U^V}$ and $Q_{Rx}^V/\mathrm{U^V}$, respectively, where Q_{Sx}^V and Q_{Rx}^V are the expected numbers of waiting tasks on tier x and $\mathrm{U^V}$ is the expected utilization of each shuttle. Note that all the tasks in the current task queues are in-tier tasks because tier-transfer tasks are only assigned to idle shuttles, while the current task in service can either be an in-tier task or a tier-transfer task targeting at tier x. Thus, this figure applies to both tier-captive configurations and tier-to-tier configurations.

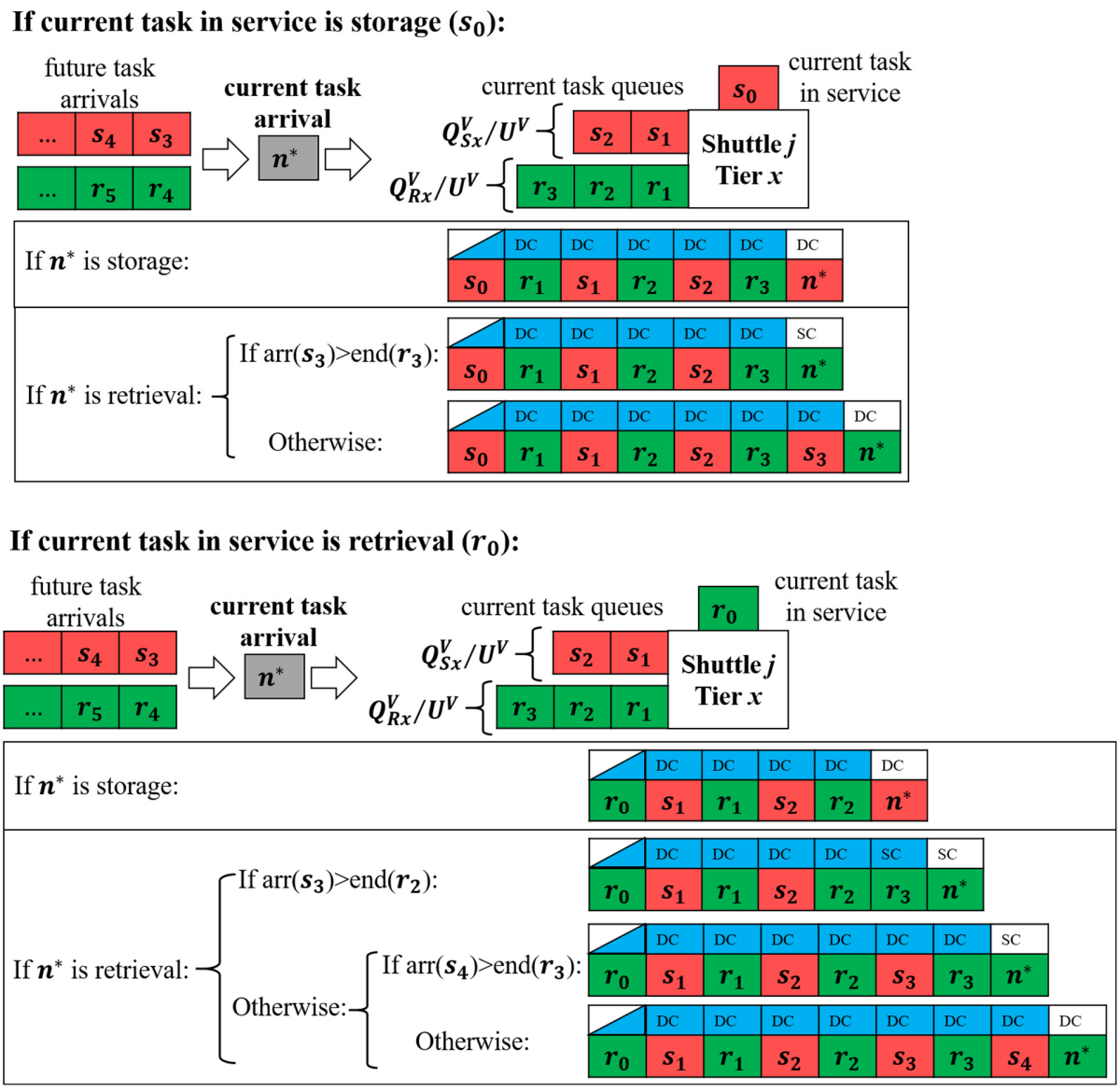

Figure 4.14 Shuttle task waiting times in Dual Cycle Mode

In Figure 4.14, all the waiting time situations are presented when the shuttle is busy. For example, if the current task in service is storage (s_0) and the current arrival task n^* is also storage, n^* will be served prior to the retrieval task r_3 even though it arrives after the arrival time of r_3 due to the DC scheduling rule – in this case, the waiting time of n^* equals the time the shuttle completes two dual-cycles ($[r_1 \rightarrow s_1]$ and $[r_2 \rightarrow s_2]$) plus the remaining service time of s_0, and n^* itself will be an DC task (because its previous task r_2 is of the different task type). On the other hand, if the current task in service is storage (s_0) and n^* is retrieval, n^* has to wait in queue until

the shuttle completes serving all previous tasks in DC patterns, and two of the following cases may happen: if the next storage task arrival (s_3) occurs after the completion of task r_3, task n^* will be served right after r_3, and its waiting time equals the service time sum of two dual cycles and a single-cycle retrieval task plus the remaining service time of s_0; otherwise, task s_3 will be served prior to task n^* even though its arrival time is later than that of n^* due to the DC scheduling rule, and the waiting time of n^* equals the service time sum of three dual cycles plus the remaining service time of s_0. An important observation here is that, if the waiting queue length of one task type (either Q_{Sx}^V or Q_{Rx}^V) is larger than the other task type, then the waiting time for shuttle service of this task type is expected to be longer. Based on the previous assumptions made in estimating for Q_{So}^V and Q_{Ro}^V, there is:

$$\frac{Q_{Sx}^V}{Q_{Rx}^V} = \frac{Q_S^V}{Q_R^V} = \frac{{cov_{aS}^V}^2 + {cov_{sS}^V}^2}{{cov_{aR}^V}^2 + {cov_{sR}^V}^2}$$

The above equations bring some insights on system design and control. As discussed previously, cov_{aS}^V and cov_{aR}^V have equal lower bounds, and both approximate 1 when the shuttle quantity is large. Reducing the variance of the external arrival sources does not help much here, when the storage assignment to tiers is random with equal probabilities (assuming COL or PRS only apply within each tier). For 1-deep racks using COL-type storage policies, cov_{sS}^V is generally slightly larger than cov_{sR}^V, because retrieval service times have less variation due to the fact that retrievals occurs randomly with equal probabilities to inventory totes. However, when the rack is 2-deep, cov_{sRi}^V is usually larger because the dual-cycle operations introduce even larger variation to the retrieval service times, as a result Q_R^V is typically slightly larger than Q_S^V for 2-deep configurations.

By studying the waiting time situations illustrated in Figure 4.14, an approximation method is proposed here to estimate the waiting times of in-tier tasks according to the relative sizes of Q_{So}^V and Q_{Ro}^V. When the shuttle is idle, the waiting time of n^* is obviously zero. When the shuttle is busy, the probability that the task currently being served is storage or retrieval can be estimated as $T_S^V/2T^V$ and $T_R^V/2T^V$, respectively, where T_S^V and T_R^V are average shuttle service times of the storage/retrieval tasks and T^V is the overall average shuttle service time computed previously. Thus, when a shuttle is busy, the expected remaining service time of its current task can be estimated as $0.5T_S^V(T_S^V/2T^V) + 0.5T_R^V(T_R^V/2T^V) = \left(T_S^{V\,2} + T_R^{V\,2}\right)/4T^V$. Denote t_{DC} as the

average dual-cycle time, and t_{SCS} and t_{SCR} are average single-cycle times of storage tasks and retrieval tasks, respectively. Based on the means and probabilities of the shuttle service cases illustrated previous, they are estimated as followed:

$$t_{DC} = t_1 + (1-\theta^R)t_9 + \theta^R t_{10};\ t_{SCS} = t_3;\ t_{SCR} = (1-\theta^R)t_5 + \theta^R t_6$$

Where t_1 is for the R-S case, t_9 for S-R, t_{10} for S-Re-R, t_3 for S-S, t_5 for R-R and t_6 for R-Re-R – these are the six in-tier service cases. Assuming the shuttle is currently on the target tier of the arriving in-tier task (thus situation 1 from Figure 4.13), and depending on the type of the arrival task (storage or retrieval) the waiting time of an in-tier task are estimated as followed:

$$W_{Sx}^V \approx \begin{cases} Q_{Sx}^V t_{DC} + \dfrac{U^V\left(T_S^{V\,2} + T_R^{V\,2}\right)}{4T^V}, & if\ Q_{Sx}^V < Q_{Rx}^V \\ Q_{Rx}^V t_{DC} + (Q_{Sx}^V - Q_{Rx}^V)t_{SCS} + \dfrac{U^V\left(T_S^{V\,2} + T_R^{V\,2}\right)}{4T^V}, & otherwise \end{cases}$$

$$W_{Rx}^V \approx \begin{cases} Q_{Sx}^V t_{DC} + (Q_{Rx}^V - Q_{Sx}^V)t_{SCR} + \dfrac{U^V\left(T_S^{V\,2} + T_R^{V\,2}\right)}{4T^V}, & if\ Q_{Sx}^V < Q_{Rx}^V \\ Q_{Rx}^V t_{DC} + \dfrac{\mathrm{U}^{\mathrm{V}}\left(T_S^{V\,2} + T_R^{V\,2}\right)}{4T^V}, & otherwise \end{cases}$$

Note that the above equations only describe the waiting times for situation 1 from Figure 4.13. With tier-captive configurations, because every task is an in-tier task and the service shuttle is always on the task's target tier (thus always in situation 1), the task waiting times are simply $W_S^V = W_{Sx}^V$ and $W_R^V = W_{Rx}^V$. For tier-to-tier configurations, however, an in-tier task may belong to either situation 1 or situation 3 (where the shuttle is not yet on the target tier), and the corresponding probabilities are J/X and $(X-J)/X$, respectively. In tier-to-tier configurations, the expected waiting times of overall in-tier tasks (both situations) are estimated as:

$$W_{Si}^V \approx \frac{J}{X}W_{Sx}^V + \frac{X-J}{X}\left[W_{Sx}^V + \max\left(W_{TT}^V - \frac{J}{2\lambda}, 0\right)\right]$$

$$W_{Ri}^V \approx \frac{J}{X}W_{Rx}^V + \frac{X-J}{X}\left[W_{Rx}^V + \max\left(W_{TT}^V - \frac{J}{2\lambda}, 0\right)\right]$$

In the above formulas, the component $\max(W_{TT}^V - J/2\lambda\,, 0)$ for both task types indicates in situation 3 – the shuttle is not yet transferred to the target tier, but this task is still an in-tier task

because it will be served after another tier-transfer task on this tier – the extra waiting time of this task for the shuttle's arrival to the target tier. When a situation-3 in-tier task arrives to the aisle, there must exist one tier-transfer task on the same target tier which is not responded by the shuttle yet. Thus, the additional waiting time of this task is difference of the waiting time of that tier-transfer task (W_{TT}^V) and the inter-arrival time of the current task after the arrival of the former ($J/2\lambda$).

Finally, the overall task waiting times are estimated as the weighted sums of the waiting times of in-tier tasks (W_{Si}^V or W_{Ri}^V) and tier-transfer tasks (W_{TT}^V), as followed:

$$W_S^V = (1-\theta^T)W_{Si}^V + \theta^T W_{TT}^V$$

$$W_R^V = (1-\theta^T)W_{Ri}^V + \theta^T W_{TT}^V$$

Inputs for the Retrieval Lift Estimation

At last, to estimate the inter-arrival time σ_a^{RL} of the retrieval lift, the retrieval inter-departure time of each shuttle σ_{dR}^V need to be estimated. Based on the approximation solutions for Two-stage Tandem Queues by Rosenshine and Chandra (1975):

$$\sigma_d^2 \approx 1/n\lambda^2 + (n-1)/n\mu^2 + (1-\rho)(n-1)/mn\mu^2 - (m-1)/m\mu^2$$
$$+ 0.5(1-\rho)(m-1)(n-1)/m^2n\mu^2 + 2(1-\rho)(m-1)(n-1)/mn^2\mu^2$$

Replace λ, μ, ρ, m and n in the above equation with $2\lambda/J$, $1/T^V$, U^V, cov_s^V and cov_a^V, respectively to estimate σ_{dR}^V. Because all retrieval tasks depart from each shuttle are going to be served by a single retrieval lift, the retrieval lift's inter-arrival time is approximated as $\sigma_d^{SL} \approx \sigma_{dR}^V/J$. The task waiting time of the retrieval lift W^{RL} is then computed according to the queuing approaches described in Section 4.5.1.

4.5.3 Finalize Analytical Results

Using the previously described approaches, the device utilizations U^V, U^{SL}, U^{RL} and U^{VL} are estimated for the shuttles (vehicles), the storage lift, the retrieval lift, and the shuttle (vehicle) lift, respectively. The average service times T^{SL} and T^{RL}, and average waiting times W^{SL} and W^{RL} are estimated for both tote lift types. Shuttle average service time T_S^V and T_R^V and average waiting times W_S^V and W_R^V are estimated for both task types.

Finally, the cycle times of storage tasks and retrieval tasks are estimated as:

$$CT_S = W^{SL} + T^{SL} + W_S^V + T_S^V$$

$$CT_R = W_R^V + T_R^V + W^{RL} + T^{RL}$$

4.6 Validation with Simulation Experiment

4.6.1 Experiment Settings

In this section, the accuracy and precision of the travel time model will be examined through simulation-based experiments. In all systems to be examined, the tier-interval and column-interval are 500mm and 550mm, respectively. The maximum velocity, acceleration, and L/U times of each device type are listed in Table 4.5.

Table 4.5 S/R devices performance parameters in validation experiment

Device Type	Max Velocity (m/s)	Acceleration (m/s^2)	Load/unload Time (s)
Tote Lift	4	6	1.75
Shuttle	4	1.5	4 (I/O; 1-deep); 6 (2-deep)
Shuttle Lift	3	3	5

As listed in Table 4.6, 25 rack designs with different capacities and different aspect ratios (presented as number of columns / number of tiers) will be examined. For each rack design, the system is further examined under four levels of vehicle deployment rate, approximately 33%, 50%, 67% and 100% of the number of tiers, respectively – a system is a tier-captive system when vehicle deployment rate is 100%, otherwise it is a tier-to-tier system. Thus, there are totally 25×4 = 100 system configurations. All of the designs are 2-deep (Z=4), and the tote lift capacity is 2. The control assumptions are COL for storage assignment and DC for shuttle scheduling. Finally, the throughput of each system configuration is estimated by the travel time model. In this experiment, the throughput of a system is assumed as its overall service rate when the utilization of any device type reached 90%.

Table 4.6 Rack designs in validation experiment

Rack Design #	**Rack Design $[X, Y, Z]$**	**Rack Capacity $= XYZ$**	**Aspect Ratio Y/X**	**Vehicles Deployed J**				**Estimated Throughput (per hr.)**			
1	[10, 40, 4]	1600	4	3	5	7	10	*164*	*320*	*496*	*628*
2	[10, 80, 4]	3200	8	3	5	7	10	*126*	*249*	*394*	*628*
3	[10, 120, 4]	4800	12	3	5	7	10	*103*	*204*	*327*	*588*
4	[10, 160, 4]	6400	16	3	5	7	10	*86*	*175*	*280*	*514*
5	[10, 200, 4]	8000	20	3	5	7	10	*74*	*153*	*244*	*456*
6	[12, 48, 4]	2304	4	4	6	8	12	*218*	*372*	*541*	*604*
7	[12, 96, 4]	4608	8	4	6	8	12	*163*	*289*	*419*	*604*
8	[12, 144, 4]	6912	12	4	6	8	12	*131*	*239*	*347*	*604*
9	[12, 192, 4]	9216	16	4	6	8	12	*111*	*203*	*297*	*560*
10	[12, 240, 4]	11520	20	4	6	8	12	*96*	*177*	*259*	*493*
11	[14, 56, 4]	3136	4	5	7	9	14	*277*	*431*	*583*	*583*
12	[14, 112, 4]	6272	8	5	7	9	14	*211*	*332*	*453*	*583*
13	[14, 168, 4]	9408	12	5	7	9	14	*171*	*269*	*370*	*583*
14	[14, 224, 4]	12544	16	5	7	9	14	*143*	*228*	*314*	*583*
15	[14, 280, 4]	15680	20	5	7	9	14	*124*	*197*	*272*	*522*
16	[16, 64, 4]	4096	4	5	8	11	16	*265*	*487*	*564*	*564*
17	[16, 128, 4]	8192	8	5	8	11	16	*197*	*369*	*540*	*563*
18	[16, 192, 4]	12288	12	5	8	11	16	*158*	*297*	*437*	*564*
19	[16, 256, 4]	16384	16	5	8	11	16	*131*	*249*	*367*	*563*
20	[16, 320, 4]	20480	20	5	8	11	16	*113*	*214*	*317*	*547*
21	[18, 72, 4]	5184	4	6	9	12	18	*325*	*540*	*546*	*546*
22	[18, 144, 4]	10368	8	6	9	12	18	*240*	*402*	*546*	*546*
23	[18, 216, 4]	15552	12	6	9	12	18	*190*	*322*	*453*	*546*
24	[18, 288, 4]	20736	16	6	9	12	18	*158*	*268*	*377*	*546*
25	[18, 360, 4]	25920	20	6	9	12	18	*135*	*230*	*324*	*545*

With each of the 100 configurations, 6 demand levels are tested, with task arrival rate λ set at 50%, 60%, 70%, 80%, 90% and 100% of the estimated throughput, respectively (note that the arrival rate of storage tasks and retrieval tasks are both λ). In this experiment, the coefficients of variation of inter-arrival times for storage tasks and retrieval tasks are consistently 1.0, which means both of the standard deviations of inter-arrival times σ_S and σ_R equal to $1/\lambda$ in all demand levels. Finally, in order to validate the analytical results, 10 simulation replications are conducted for each configuration-demand scenario, each with 500-hour running length plus 100-hour warm-up period, thus the total number of simulation runs are 100×6×10 = 6000. The system control policies in the simulation model is consistent with the corresponding control assumptions of the

travel-time model. The task arrival rates will slightly vary around the default values to control the inventory levels by the same method introduced in Section 4.4.6.

4.6.2 Experiment Results

Travel Time Model Outputs

Three types of outputs from the travel time model are viewed as performance indicators for each system design and configuration under each demand scenario. First of all, the utilization outputs U^{SL}, U^{RL}, U^{V} and U^{VL} are examined to measure the quality of system throughput estimate. Secondly, the cycle time outputs CT_S and CT_R for storage tasks and retrieval tasks, respectively are examined to measure the quality of system responsiveness estimates. In addition, the three critical coefficients: relocation ratio θ^R, dual-cycle ratio θ^D and tier-transfer ratio θ^T are examined in order to evaluate the validity of the queuing-network analysis approaches in the travel-time model.

Some initial interpretations can be made by observing the analytical results. In Figure 4.15, the estimates for the maximum throughputs of the 25 rack designs (obtained when all tiers are deployed with shuttles, thus $J = X$) are categorized by the number of tiers (X) and aspect ratios (Y/X) of the designs. The throughput curves for $Y/X = 4$ and $Y/X = 8$ completely overlap because the bottlenecks of such short rack designs are the tote lifts – as shown in Figure 4.17, the tote lift utilizations (because $U^{SL} \approx U^{RL}$, we did not display them separately here) reached 90% in all these designs regardless of the number of tiers. As the number of tiers increase, each shuttle utilization curve decreases (Figure 4.16), while each tote lift utilization curve (Figure 4.17) increases until it reaches 90%, indicating that the tote lift utilizations become more constraining (of the throughput) as the racks get taller (in this experiment, we use the terms "tall" and "short" to describe the number of tiers X, and the terms "deep" and "shallow" to describe the number of columns Y). In Figure 4.15, all the throughput curves merge at $X = 18$ – which means with 18 tiers, the tote lifts become the bottleneck of the entire aisle for all the aspect ratios under consideration.

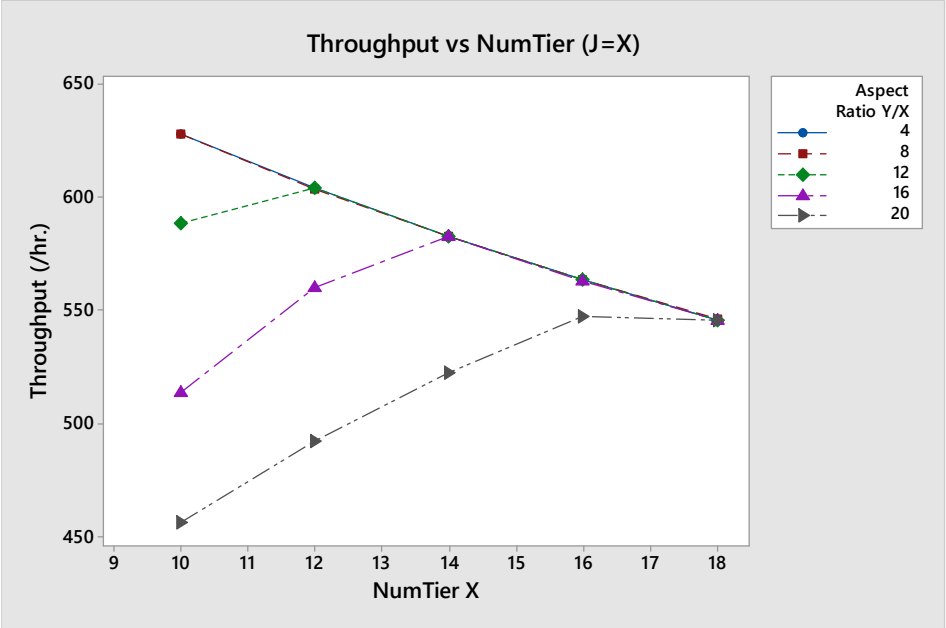

Figure 4.15 Estimated maximum throughputs ($J = X$) for the rack designs

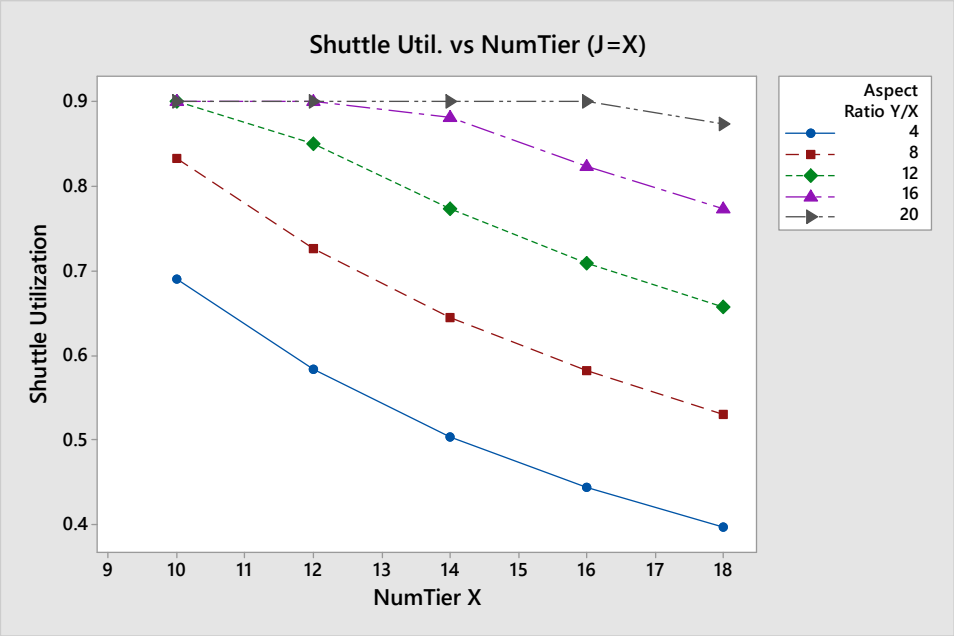

Figure 4.16 Estimated shuttle utilizations of the rack designs at maximum throughputs

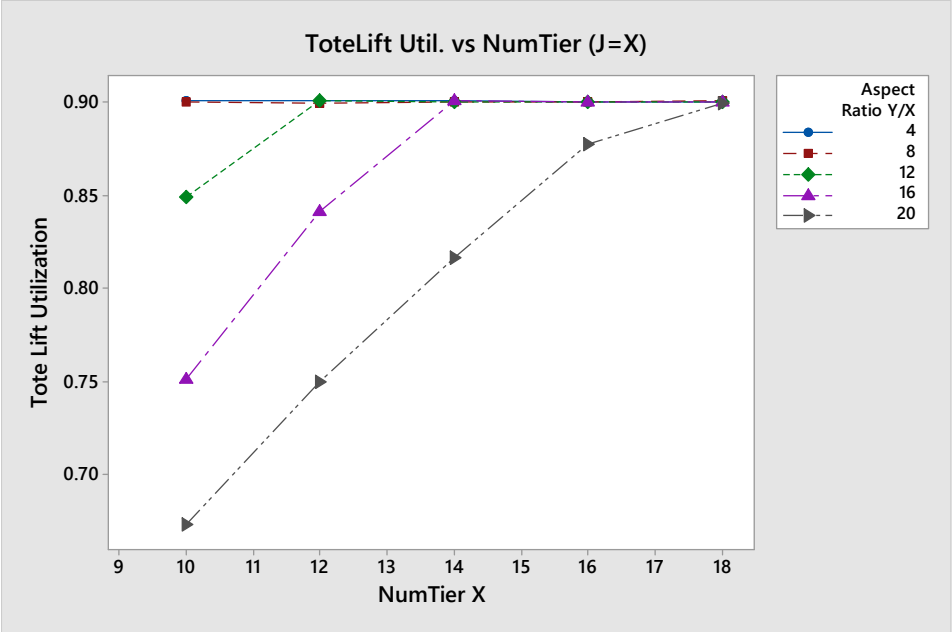

Figure 4.17 Estimated tote lift utilizations of the rack designs at maximum throughputs

The above result interpretations illustrate the maximum throughputs of the rack designs when all tiers have tier-captive shuttles. It can be inferred that the shape of the rack has significant and complex impact to the maximum throughput of the design: in general, the tote lifts will become the bottleneck when the rack is taller and shallower, while the shuttles will become the bottleneck when the rack is shorter and longer – which matches common intuition. In addition, the device utilizations at maximum throughput to some extent indicate the potential effects of tier-to-tier configurations (thus not deploying the full number of shuttles): for example, with the $X = 18, Y/X = 4$ design (tall and shallow) where the utilizations are 90% for the tote lifts but only 40% for the shuttles, it can be assumed that reducing the number of shuttles will not decrease the aisle's throughput until this number reached a lower bound (however, that will increase the task cycle times).

Take the 18-tier designs as an example to further illustrate the results from its tier-to-tier configurations. There are totally 20 configurations here: four shuttle deployment levels for each of the five aspect ratios. Figure 4.18 shows the estimated throughputs of each rack design when different numbers of shuttles are deployed. It can be inferred that for $Y/X = 4$ and 8 designs (tall and shallow racks), 12 shuttles are adequate to maintain the maximum throughputs of the aisles. However, as shown in Figure 4.19, increasing the number of shuttles from 12 to 18 is expected to

significantly reduce the retrieval tasks' cycle times for both designs, although not improving their throughputs.

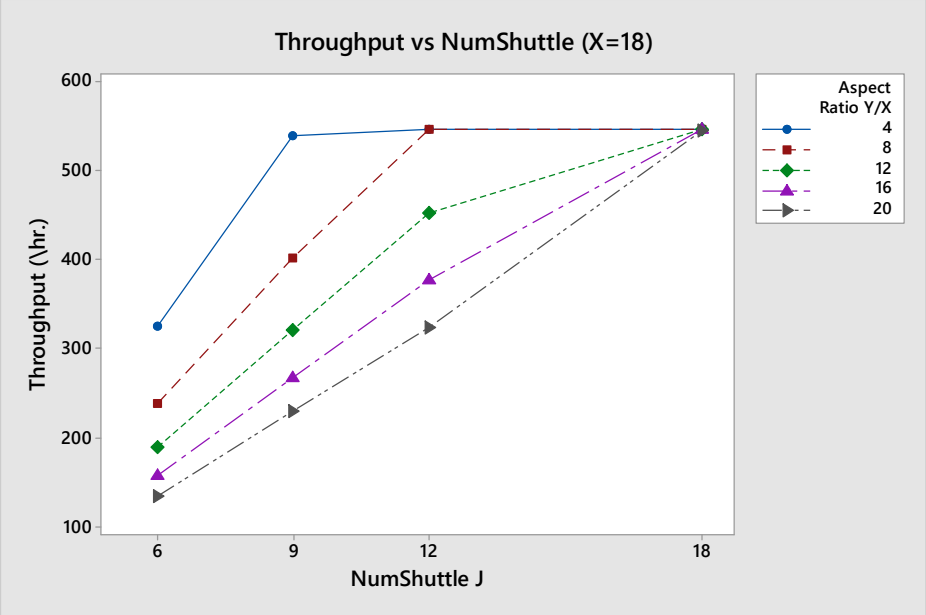

Figure 4.18 Estimated throughputs of shuttle configurations

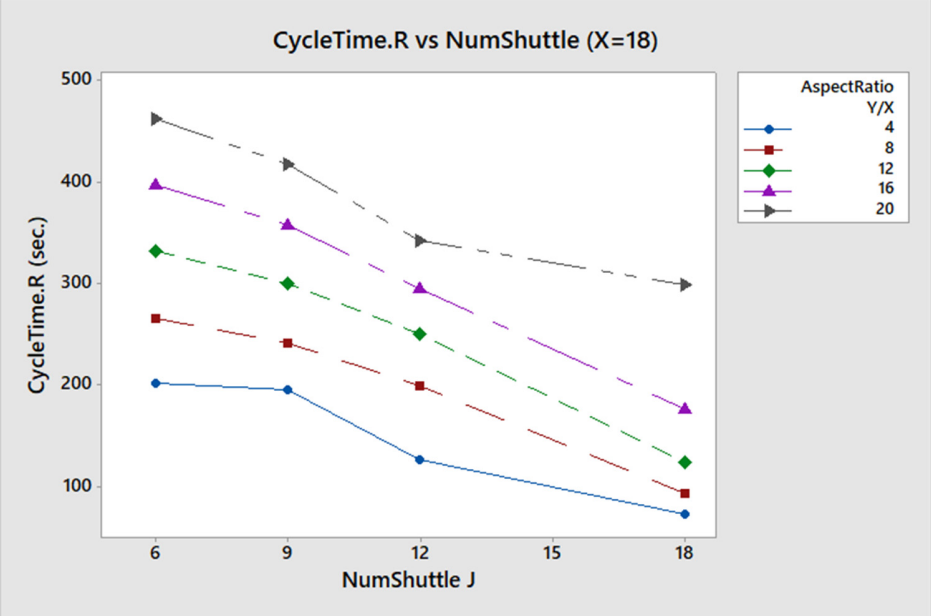

Figure 4.19 Estimated cycle times (retrieval) at throughput levels of shuttle configurations

At last, the estimated relocation ratio θ^R, dual-cycle ratio θ^D and tier-transfer ratio θ^T (at throughput levels) of the 18-tier designs are illustrated in Figure 4.20. Relocation ratio θ^R is a constant: as explained in Section 4.4 given the storage policy, it is only determined by the rack utilization distribution assumption. As illustrated in the approximation approaches in Section 4.5.2, θ^D and θ^T are complex functions. In general, the tier-transfer ratio θ^T gets smaller when there are more shuttles deployed in the aisle, and its upper bound is $X - J/X$. The dual-cycle ratio θ^D typically gets larger when the shuttle utilization gets higher, and its lower bound is 0.5. Generally speaking, high θ^D and low θ^T are favorable and probably indicate good system designs or/and shuttle configurations.

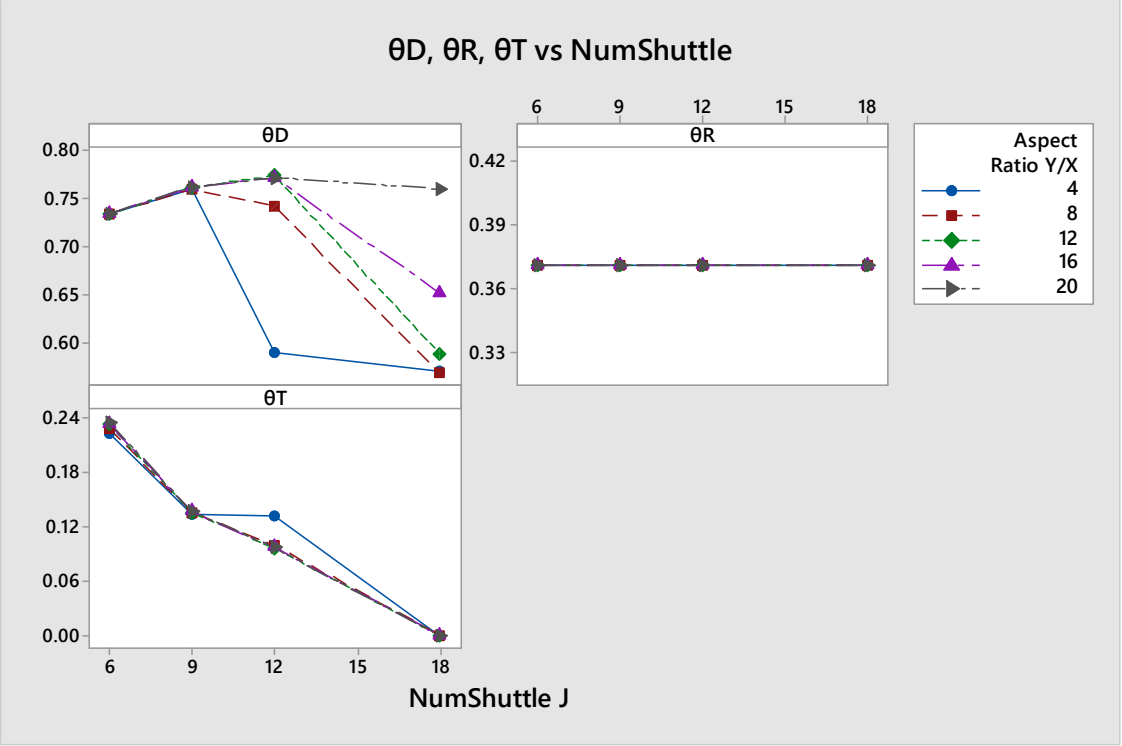

Figure 4.20 Estimated relocation ratio θ^R, dual-cycle ratio θ^D and tier-transfer ratio θ^T at throughput levels of shuttle configurations

Analytical Results Quality Interpretations

The quality of the travel time model results is measured as the differences between its analytical outputs and the simulation outputs. For each performance indicator I, its error I_E is presented as $I_E = (I_S - I_A)/I_S$, where I_A is analytical estimate and I_S is simulation output. For

example, if in a particular configuration and scenario the estimated utilization of the shuttle lift is 44% and the simulation result (average over 10 simulation replications) is 40%, then $I_E(U^{VL}) = -10\%$.

Table 4.7 shows the validation results in terms of the absolute estimation errors $|I_E|$. The results are categorized by the indices of demand scenarios – for each of the 100 system configurations, scenario 6 is corresponding to the arrival rates λ_{max} at the throughput levels, and scenarios 1 to 5 are corresponding to the arrival rates equals to 50%, 60%, 70%, 80% and 90% of the λ_{max}, respectively. Based on analytical and simulation results, the absolute errors of the nine indicators are computed for each configuration-scenario (totally 600 combinations). Then, the averages of the absolute errors are computed for each scenario index. The results are listed separately for tier-captive configurations ($J = X$) and tier-to-tier configurations ($J < X$) because the latter is based on a more complex queuing model.

Table 4.7 Travel time model average estimation errors (absolute) by demand scenarios

SCE.	**Absolute error $\|I_E\| = \|I_S - I_A\|/I_S$** [Tier-captive (1 shuttle deployment level), Tier-to-tier (average from 3 shuttle deployment levels)]								
	U^{SL}	U^{RL}	U^{V}	U^{VL}	CT_S	CT_R	θ^D	θ^R	θ^T
1	[2.3%, 2.0%]	[0.6%, 0.4%]	[1.0%, 2.0%]	[N/A, 5.5%]	[6.2%, 8.3%]	[6.7%, 8.0%]	[7.1%, 1.8%]	[1.7%, 1.7%]	[N/A, 5.5%]
2	[2.0%, 2.1%]	[0.8%, 0.5%]	[0.9%, 1.6%]	[N/A, 4.4%]	[5.4%, 8.6%]	[6.8%, 8.7%]	[4.4%, 2.8%]	[1.8%, 1.7%]	[N/A, 4.4%]
3	[1.5%, 2.1%]	[0.8%, 0.6%]	[0.8%, 1.2%]	[N/A, 3.5%]	[3.9%, 7.6%]	[6.2%, 7.9%]	[4.0%, 3.2%]	[1.7%, 1.7%]	[N/A, 3.5%]
4	[1.0%, 2.1%]	[0.9%, 0.7%]	[0.7%, 1.1%]	[N/A, 3.7%]	[2.1%, 6.1%]	[4.9%, 6.3%]	[3.4%, 2.0%]	[1.7%, 1.7%]	[N/A, 3.5%]
5	[0.6%, 2.0%]	[0.9%, 0.8%]	[0.6%, 1.3%]	[N/A, 6.3%]	[2.1%, 5.7%]	[3.4%, 4.9%]	[2.4%, 3.0%]	[1.7%, 1.7%]	[N/A, 5.8%]
6	[0.2%, 1.9%]	[0.8%, 0.8%]	[0.6%, 1.9%]	[N/A, 13.6%]	[7.2%, 11.2%]	[3.8%, 9.7%]	[2.2%, 8.9%]	[1.8%, 1.7%]	[N/A, 12.9%]
AVG	**[1.3%, 2.0%]**	**[0.8%, 0.6%]**	**[0.8%, 1.5%]**	**[N/A, 6.2%]**	**[4.5%, 7.9%]**	**[5.3%, 7.6%]**	**[3.9%, 3.6%]**	**[1.7%, 1.7%]**	**[N/A, 5.9%]**

It can be observed that the utilization estimates for the storage lift (U^{SL}), the retrieval lift (U^{RL}) and the shuttles (U^{V}) are reasonable – the error is within 2% in general. This is strong evidence that the travel-time model is valid for throughput estimation. Also, the estimation for the three critical coefficients: relocation ratio θ^R, dual-cycle ratio θ^D and tier-transfer ratio θ^T, are also reasonable. Although these coefficients do not directly affect the throughput or responsiveness

indicators, good estimation of them is more evidence of the validity of the techniques applied in the analytical modeling processes – otherwise, it would be very difficult to examine the model given its complexity. The estimates for shuttle lift utilization (U^{VL}) at higher demand levels (scenario 6) have larger error for tier-to-tier configurations, but that will not affect the throughput estimation because U^{VL} is generally small, and according to the control assumptions made previously the shuttle lift will never become the bottleneck of the system because there is always $U^{VL} < U^{V}$. Similarly, the relatively high error for θ^{T} at high demand levels is not concerned as a significant problem: as illustrated previously, θ^{T} is typically small when the demand levels are high. (In fact, it is observed that the estimation error of θ^{T} have strong positive correlation with that of U^{VL}, which is believed as originated from the approximation techniques in obtaining θ^{T}. We view this as a future opportunity to improve the model. Thus, the error of U^{VL} is not considered in evaluating the quality of the throughput estimates (however, just like the three ratios, U^{VL} is still viewed as one indicator of the validity of the analytical model). At the highest demand levels (Scenario 6, under which at least one device utilization reached 90% for each design configuration, and thus the corresponding task arrival rate λ is viewed as the throughput of the design configuration), the estimation errors of the maximum utilizations among all devices (excl. shuttle lifts') are 1.3% for tier-captive configurations and 2.0% for tier-to-tier configurations – although the device utilizations are just intermediate measure of the throughputs, we interpret these results as indicating that the estimation precisions of the throughput performance are in general over 98.7% and 98.0%, respectively.

The estimation quality for the systems' responsiveness indicators are presented as the absolute errors of the cycle time estimates. With the tier-captive configurations, the average estimation errors for task cycle times are 4.5% and 5.3% for storage tasks and retrieval tasks, respectively. The overall estimation precision for the cycle times are thus viewed as $100\% - (4.5\% + 5.3\%)/2 = 95.1\%$. For the tier-to-tier configurations, these numbers are 7.9% and 7.6% for storage tasks and retrieval tasks, respectively, and the overall precision is 92.2%. Recall that the system is a tandem queuing system with inter-leaving entities, and that single-server and multi-server queuing system characteristics both exist in the system, estimating the task waiting times at different service stages is very difficult, and over 90% precision in general is concerned as acceptable for the cycle times.

Further Exploration of Estimation Errors

As discussed before, the quality of throughput estimation is viewed as satisfactory. Here we further explore the errors observed in the cycle time estimates. Figure 4.21 illustrates the estimation errors of retrieval cycle times CT_R – the estimation errors of storage cycle times CT_S have very similar patterns to those of CT_R, thus are not further demonstrated here. The error data points include all 100 system configurations: 25 designs each with three levels of tier-to-tier configurations and one tier-captive configuration, in which configurations #1, 5…4(n-1)+1…97 are tier-to-tier configurations with shuttle deployment level 1 for designs #1, 2…n…25 (as listed in Table 4.6), respectively, configurations #4, 8…4(n-1)+4…100 are tier captive configurations (shuttle deployment level 4), etc. All the data points are categorized by six demand scenarios, where 6 is the highest demand level. Figure 4.22 and Figure 4.23 further display the error data separately for the 25 tier-captive configurations (150 data points) and the 75 tier-to-tier configurations (450 data points). Among the 600 data points the errors range from -25.1% to +35.7%, where a negative error indicates overestimation and a positive error indicates underestimation (with respect to simulation results). Obviously, the error range for the tier-captive configurations ([-7.9%, 14.7]) are in general much smaller than that for the tier-to-tier configurations ([-25.1%, 35.7%]). The average of the 600 data points is 3.0% (4.7% for tier-captive, 2.4% for tier-to-tier), and the average of their absolute values is 7.0% (5.3% for tier-captive, 7.6% for tier-to-tier).

The error estimation for tier-captive configurations is viewed as satisfactory. For tier-to-tier configurations, it can be observed from the chart that when the demand levels are high, the model tend to overestimate the cycle times. It can also be observed that with tier-to-tier configurations, the cycle times of systems with taller racks (configurations with larger ID) are in general underestimated when the demand levels are low. Furthermore, the shuttle deployment levels appear to have significant effects on the errors. Thus, we select the following four factors and analyze their effects to the estimation error of CT_R:

1. Demand Scenario: 1 to 6 indicate λ corresponding to 50%, 60%, 70%, 80%, 90% and 100% of the estimated throughput, respectively;
2. Number of tiers X of the rack;

3. Aspect ratio Y/X between number of columns Y and number of tiers X: smaller and larger aspect ratios indicate shallower and deeper racks, respectively;
4. Shuttle deployment level in the rack: 1, 2, 3 corresponding to J = 33.3%, 50% and 66.7% of X (round up/down to the closest integers).

The main effects of the four selected factors are demonstrated in Figure 4.24. Consistent with the previous observations, in general CT_R is overestimated at the highest demand level (where λ = throughput) – we further observed that it is highly related to the overestimation of the tier-transfer ratio θ^T at high demand levels when the approximations for the virtual TT system become less precise. Also, CT_R appears to be underestimated for tall rack systems in which the tote lifts are more burdened. The error of CT_R is also affected by the aspect ratios and the shuttle deployment levels, while the direct effects from all four factors are relatively minor (with in [-10%, 10%]). Interaction effects of the factors need to be further analyzed.

Figure 4.25 demonstrates the interaction effects of the four selected factors. Strong interaction is observed between the number of tiers and the shuttle deployment levels: when the deploy level is low (level 1), CT_R is tend to be underestimated for taller racks (e.g. $X = 18,\ J = 6$) but overestimated for shorter racks (e.g. $X = 10,\ J = 3$). We assume this is again related to the approximation of the virtual TT system – because it is approximated as a bounded M/M/c system where $c = J$, the task waiting times might be underestimated with larger J's and overestimated with smaller J's. Together with the effects from the other factors, such interaction effects lead to the -25.1% lower bound and +37.1% upper bound of the estimation errors for CT_R. These observations provide insights for further improving the validity of the travel time model.

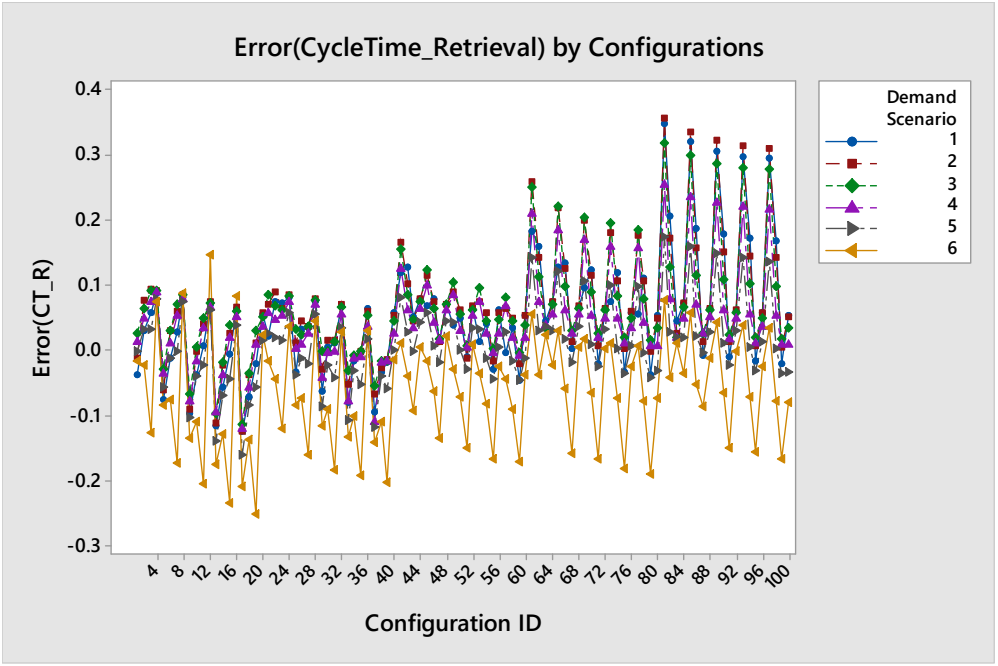

Figure 4.21 Estimation Error of CT_R by System Configurations and Demand Scenarios

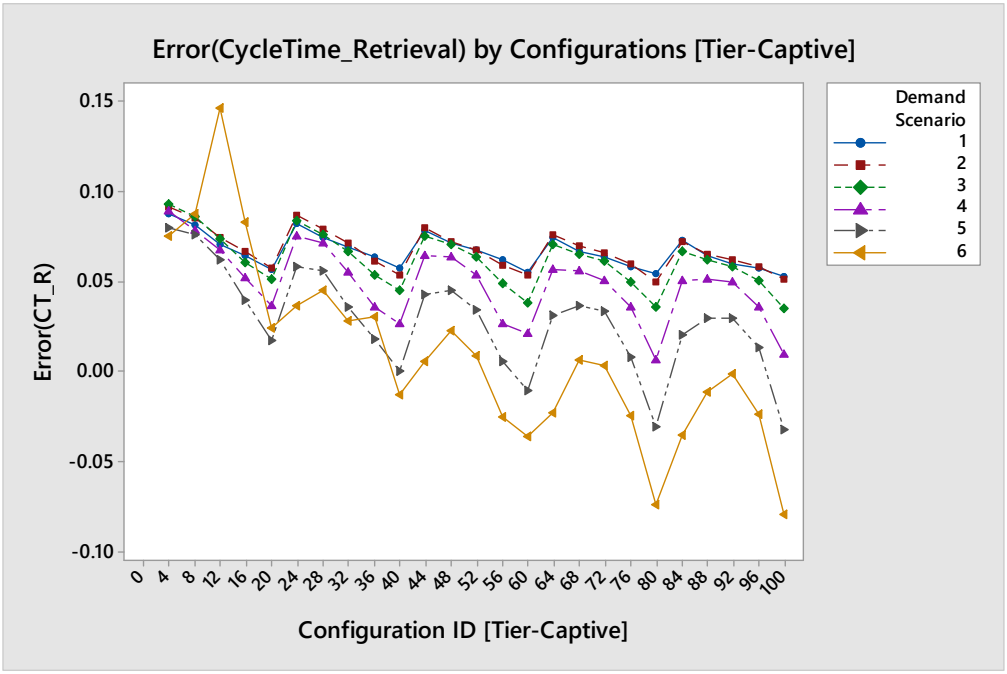

Figure 4.22 Estimation Error of CT_R by System Configurations and Demand Scenarios (Tier-captive)

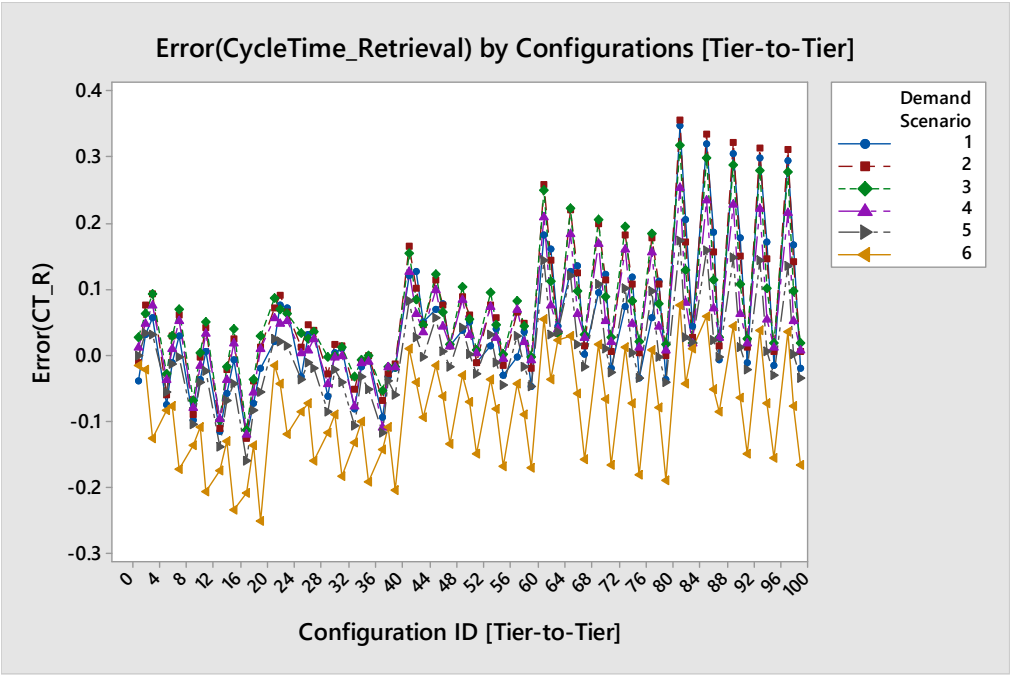

Figure 4.23 Estimation Error of CT_R by System Configurations and Demand Scenarios (Tier-to-tier)

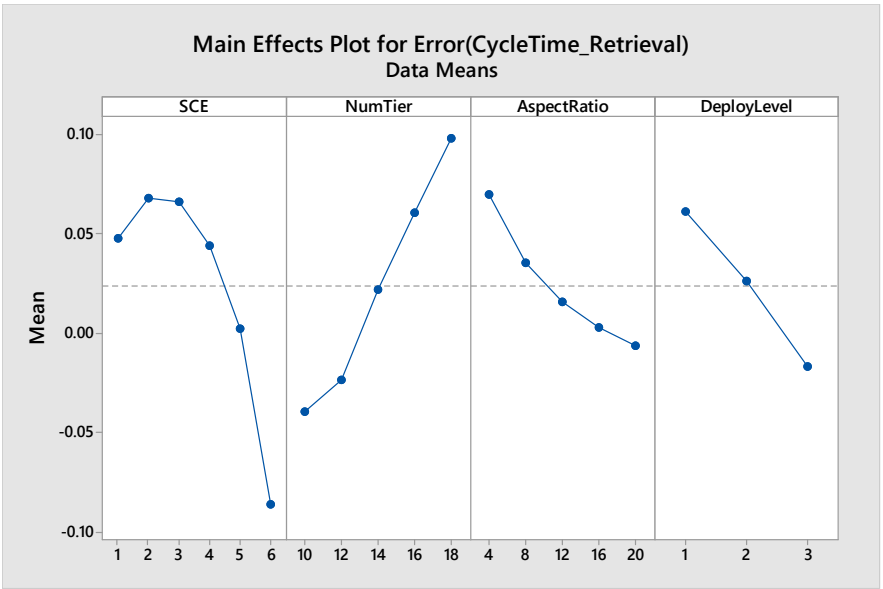

Figure 4.24 Main Effects for Estimation Error of CT_R (Tier-to-tier)

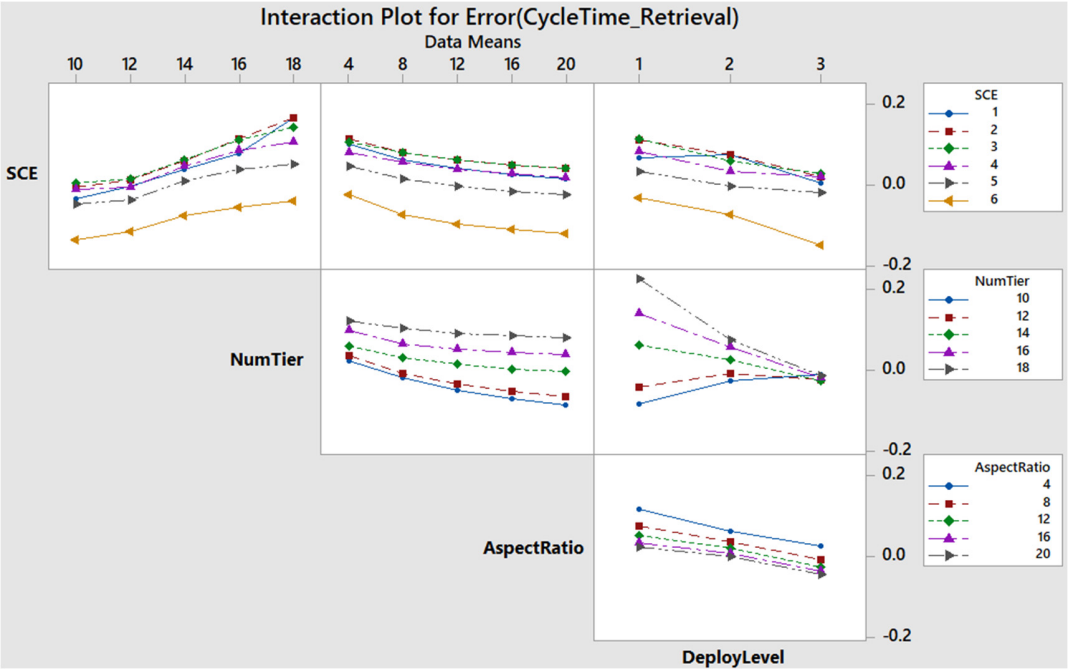

Figure 4.25 Interaction Effects for Estimation Error of CT_R (Tier-to-tier)

In conclusion, the travel time model of the analytical approach is considered as acceptable based on the simulation-based validation experiment results. Both the throughput estimates and the responsiveness estimates are viewed as with adequate preciseness for conceptual design purpose. To further validate the analytical techniques applied, critical intermediate results from the travel time model are also examined by comparing to corresponding simulation outputs, and the results are satisfying.

4.7 Application Example

In this section, an example of the proposed application in system design decisions is presented with regard to the analytical approach. Assume the decision-maker plans to build an SBS/RS for an E-commerce distribution center. To deal with the seasonality of the demand and possible expanding of the business in the future, the decision-maker chooses to implement tier-to-tier configurations in the system. In this example, the goal is to find design(s) that minimize the overall capital expenditures while also satisfying the space constraints and performance requirements. Because the demand may increase in the future, the decision-maker need to consider about the expandability of the system and the related costs as well.

Space Constraints

For the simplicity of illustration, all the aisles in the expected system are assumed 2-deep (thus 4 slots per column) and have identical rack designs. Also, the aisles are all tier-to-tier configurations to accommodate for future demand changes. Due to the space constraints of the distribution center, the maximum system scale is limited to 12 aisles, 25 tiers and 400 columns.

Demand Scenarios and Performance Requirements

The inter-arrival times of both types of tasks to each aisle are assumed exponential. Each task is related to an independent customer order, and each task is related to a single tote. The target rack capacity of the overall system is 200,000 slots, and the expected rack utilization is 0.8 for each aisle. The target throughput of the overall system is 3,000 storage tasks and 3,000 retrieval tasks per hour for now. In addition, because the demand may grow in the future, the system's throughput is required to be expandable up to 5,000 S/R tasks per hour by adding shuttles in the future. This can be interpreted as the system throughput when all aisles are deployed full numbers of shuttles. Finally, the expected cycle time of both task types should be less than 600 seconds.

Objective Function

The capital expenditure of an SBS/RS aisle is viewed as the sum of the lift costs (which is fixed costs per aisle), rack costs (which is a function of rack size), and the shuttle costs (which is determined by the number of shuttles purchased and deployed). In this example, the cost function of the overall system is assumed given as:

$F_{cost} = A(20{,}000 + 2{,}000\mathrm{X} + 50\mathrm{XY} + 5{,}000\mathrm{J})$ US dollars, where A is the number of aisles, X and Y are the number of tiers and number of columns per aisle, respectively, and J is the number of shuttles per aisle. We are working with our industry partner to verify these estimates, but the specific cost numbers are not critical for the demonstration of our analysis (but they would be very important when applying the methodology in a production environment).

Analytical Approach and Results Interpretation

According to the space constraints, the maximum possible rack capacity of a single aisle is 25×400×4 = 40,000, thus a minimum of 5 aisles are needed to meet the target capacity 200,000 of the overall system. Designs including 5, 6, 7…12 aisles will be evaluated. Two throughput

targets are considered in searching the designs: the required throughput THP_0 to fulfill the current demand (3,000/hr.), and the required throughput to fulfill the expected expanded demand THP_{Max} (5,000/hr.). Because the configurations are assumed identical for all aisles, the capacity targets and the throughput targets per aisle are listed in Table 4.8.

Table 4.8 Analytical approach example: Design candidates screening

#Aisles	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Capacity Target K	40000	33334	28572	25000	22223	20000	18182	16667
Throughput target THP_0	600.00	500.00	428.57	375.00	333.33	300.00	272.73	250.00
Throughput target THP_{Max}	1000.00	833.33	714.29	625.00	555.56	500.00	454.55	416.67
# Designs	1	5	8	10	12	13	14	15
# Eligible Designs	0	0	0	0	0	8	12	13
# Eligible Configs	0	0	0	0	0	85	140	155

The search problem is then simplified as evaluating single-aisle designs given different throughput requirements. All designs that satisfy $X \leq 25, Y \leq 400, XYZ \geq K$ and $X(Y-1)Z < K$ are evaluated, where X is the number of tiers, Y is the number of columns, and $Z = 4$. Thus, 78 single-aisle designs are evaluated. (Note that this number is small for illustration purposes, because the aisles are assumed identical, and the search space only includes the minimum designs that meet the capacity requirements. In practice, the designer may need to face much larger search spaces with larger capacity range and heterogeneous aisle). Then, a single-aisle design is considered eligible in meeting the demands when it meets all of the following requirements:

1. When the maximum number of shuttles are deployed, the maximum sustainable throughput is no less than $\boldsymbol{THP_{max}}$/hr**.** (in this example, "sustainable" means the maximum device utilization is less than 90%);
2. When the maximum number of shuttles are deployed and task arrivals at $\boldsymbol{THP_{max}}$/hr., the expected task cycle times are no larger than 600 seconds.

Among the 78 designs that meet the capacity requirements, 33 designs that meet the expanded throughput requirements THP_{Max} and cycle time requirements are considered as eligible – these designs are only found in the 10, 11 and 12 aisle cases. In the next step, the aisle configurations with different numbers of shuttles deployed are evaluated. 380 eligible configurations that meet the current throughput requirements THP_0 and the cycle time

requirements are found for each eligible design. Table 4.9 shows the high-level analytical results for the 33 eligible designs. For each design, the minimum numbers of shuttles J_{min} that required to meet the current and future throughput targets are listed – these are the minimum configurations of each rack design. The maximum sustainable throughputs of the designs are obtained at the maximum configurations (when $J = X$, thus every tier is deployed a shuttle). The capital expenditures of the minimum configurations for both the current demands and the future demands are computed according to the given cost function F_{cost} — it is assumed that the decision-maker will only purchase the minimum number of shuttles for now to meet the current demand, if the demand increases in the future, more shuttles will be purchased as needed. Finally, task cycle times are estimated under the minimum configurations of the current demands.

From the results, the 10-aisle, 18-tier, 278-column design requires the least capital expenditure to meet the current demands. However, the 11-aisle, 14-tier, 325-column design appears to be more economical if the future demands increase as expected. Moreover, the 12-aisle, 16-tier, 261-column design have the highest potential throughput, which indicates higher flexibility against unexpected demand surges. The criteria become more complicated if the responsiveness (cycle times) concerns are incorporated in the cost function, and the user might be interested in exploring more analytical details – for example, the user can obtain the throughput / cycle time curves plotted as functions of number of shuttles to further explore configuration performance with the presence of demand uncertainty and seasonality, and even develop a shuttle-deployment strategy – we will not discuss those in depth. The point of this example is that, our analytical approach can rapidly and systematically evaluate large number of system designs and configurations according to different design requirements and knowledge of the demands, and provide performance indicators which allow the decision-maker to flexibly develop the most appropriate evaluation criteria for his business.

Table 4.9 Analytical approach example: Design candidate evaluation

# Aisles	[X, Y]	Capacity	J_{min} (per aisle)		Max λ / hr. $(J = X)$	$F_{cost}(J = J_{min})$ ($)		Cycle Times (sec)	
			λ = 3,000	λ = 5,000		λ = 3,000	λ = 5,000	CT_S	CT_R
10	[16, 313]	200320	11	16	5555	3,574,000	3,824,000	197	225
	[17, 295]	200600	11	16	5548	3,597,500	3,847,500	177	203
	[18, 278]	200160	10	16	5461	3,562,000	3,862,000	216	245
	[19, 264]	200640	10	15	5377	3,588,000	3,838,000	196	221
	[20, 250]	200000	10	15	5297	3,600,000	3,850,000	178	202
	[21, 239]	200760	9	14	5219	3,579,500	3,829,500	233	261
	[22, 228]	200640	9	14	5143	3,598,000	3,848,000	208	233
	[23, 218]	200560	9	14	5070	3,617,000	3,867,000	191	214
11	[14, 325]	200200	10	14	5209	3,580,500	3,800,500	223	256
	[15, 304]	200640	10	15	5833	3,608,000	3,883,000	191	220
	[16, 285]	200640	10	15	6203	3,630,000	3,905,000	172	197
	[17, 268]	200464	9	14	6103	3,594,800	3,869,800	217	246
	[18, 253]	200376	9	14	6007	3,615,700	3,890,700	192	218
	[19, 240]	200640	9	13	5915	3,641,000	3,861,000	177	200
	[20, 228]	200640	9	13	5826	3,663,000	3,883,000	165	186
	[21, 217]	200508	8	13	5740	3,628,350	3,903,350	216	241
	[22, 207]	200376	8	12	5657	3,648,700	3,868,700	195	218
	[23, 198]	200376	8	12	5577	3,670,700	3,890,700	179	201
	[24, 190]	200640	8	12	5499	3,696,000	3,916,000	168	187
	[25, 182]	200200	8	12	5424	3,712,500	3,932,500	158	176
12	[13, 321]	200304	10	13	5322	3,655,800	3,835,800	162	190
	[14, 298]	200256	9	13	6017	3,619,200	3,859,200	203	233
	[15, 278]	200160	9	13	6749	3,642,000	3,882,000	178	204
	[16, 261]	200448	9	13	6767	3,669,600	3,909,600	164	188
	[17, 246]	200736	8	12	6658	3,637,200	3,877,200	211	238
	[18, 232]	200448	8	12	6554	3,657,600	3,897,600	188	212
	[19, 220]	200640	8	12	6453	3,684,000	3,924,000	173	195
	[20, 209]	200640	8	12	6356	3,708,000	3,948,000	161	181
	[21, 199]	200592	8	11	6262	3,731,400	3,911,400	152	170
	[22, 190]	200640	7	11	6172	3,696,000	3,936,000	206	230
	[23, 182]	200928	7	11	6084	3,723,600	3,963,600	190	211
	[24, 174]	200448	7	11	5999	3,741,600	3,981,600	175	195
	[25, 167]	200400	7	10	5917	3,765,000	3,945,000	163	182

4.8 Summary

In this chapter, a general analytical approach is established to support the conceptual design of SBS/RSs. System performance under evaluation is indicated by system throughput and responsiveness, interpreted in this research as device utilizations and task cycle times, respectively. In our analytical approach, system design/configuration options involving tier-to-tier configurations, different rack depths, and different tote lift capacities, etc. are included. In order to improve the precision of the estimates, both Pure-Random-Storage policy and Closest-Open-Location policy are considered. Various demand scenarios are studied, and the shuttles' dual-cycle performances are evaluated. A precise and efficient three-stage iterative analytical approach is proposed. In the first stage, the task visit probabilities of the rack slots are estimated based on the demand and storage policy assumptions. In the second stage, queuing analysis is conducted with integrated considerations for the tandem queuing and multi-server queuing characteristics of the system and approximations of relocation, dual-cycle scheduling, and tier-transfer service processes. In the third stage, the candidate design is evaluated, and the program systematically iterates the search and estimation process to find the best candidate(s).

The animated, data-driven and data-generated simulation model developed in Chapter 3 has played a very important role in the development of this analytical approach: due to the complexity of the travel time model, the assumptions and approximations made in the analytical approach are continuously improved and fine-tuned by the simulation model. At last, the travel time model is validated by Monte-Carlo experiments based on the simulation model. Experiment results show that for both tier-captive and tier-to-tier configurations, the validation results are still satisfactory in term of estimation precision of the throughput and task cycle time. The analytical approach is thus viewed acceptable as a precise and efficient tool for design evaluation purpose. Finally, techniques applied in this analytical approach are expected to be the baseline and foundation of the control strategy development – to be explored in Chapter 5.

Chapter 5 Operational Control Strategy Development

5.1 Overview

In this chapter, operational control strategies are studied aiming at optimizing the throughput and responsiveness performance of SBS/RS. Two types of control policies discussed in the previous chapter – storage assignment and device scheduling – are further explored based on the observations from the proposed analytical approach. Due to the nature of the dynamics of SBS/RS operations, the control policies are highly coupled with each other. In addition, the SKU (stock keeping unit)-level characteristics like turn-over rates and demand correlations further increase the complexity of the control problem but also bring opportunities for improvements.

Both the short-term effects and long-term effects of the control policies need to be considered – the former refers to the control performances in reducing the makespan of the current tasks in the system, and the later refers to the control performances in improving the sustainable throughput and responsiveness of the system. Thus, mathematical programming approaches and dynamic dispatching approaches are explored to improve the overall performance of the control strategies. In addition, the SKU-level characteristics are incorporated in the research scope to guide the development of the control strategies.

The control strategy development is based on the data-driven and data-generated simulation model introduced in Chapter 3. The simulation model provides a comprehensive experimental platform to simulate combinations of demand scenarios, designs and configurations, and control strategies, to facilitate system performance evaluation with critical indicators presented in numerical and graphical forms.

5.2 Operational Control Decisions in SBS/RS

In SBS/RS-based warehouse systems, both storage and retrieval tasks are performed to fulfill the requests posed by pickup and replenishment functionalities. The sequences of the tasks

form *schedules* for each device in the aisle. A *schedule* describes the sequence and timings that a specific S/R device serves tasks. With a traditional crane-based AS/RS, the crane serves complete storage and retrieval tasks so the schedule is simply a sequence of storage and retrieval tasks. However, in SBS/RS, the scheduling is more complex since the storage and retrieval tasks require multiple independent S/R devices (lifts and shuttles) with buffers in between. Lastly, the schedule of the shuttle lift is defined by a sequence of tier-to-tier operations, describing the shuttle to be selected and destination tier in each operation. As discussed in Chapter 4, those schedules are dynamic and highly coupled with each other due to the service patterns of the SBS/RS: the schedule of the storage lift breaks into storage tasks for each shuttle, while the retrieval tasks from each shuttle converge into a single schedule for the retrieval lift. Thus, delays with the first-stage device may lead to delays on the next-stage device. Also, if the tote lift capacities are larger than one, the lift may carry one or multiple totes at a time, thus tours of different sizes will occur. The problem is more complicated in tier-to-tier configurations where shuttles' tier-transfer services are introduced. Figure 5.1 illustrates the schedules of each device type and the correlations between the schedules. Also, as studied in Chapter 4, the assignment of storage locations (either for both storage totes or for blocker totes in retrieval tasks) largely affects not only the cycle times of the current storage tasks but also the cycle times of future retrieval tasks, thus affects the system's throughput and responsiveness performances. Moreover, the SKU-level characteristics like turnover rates and demand correlations further complicate the control problems, while wise control policies based on these characteristics are expected to reduce both the device travel times and the relocation times significantly, thus reduce the overall task cycle times.

A *task* describes the information required for the devices to fulfill S/R demands within the same SBS/RS aisle. Both types of tasks need to be performed by shuttles and lifts. The releasing of those tasks, scheduling of devices, and flowing of products, all need to be guided by specific *algorithms*. In the context of this research, an "*algorithm*" can be realized either through Mathematical Programming (MP) approaches or through Dynamic Dispatching (DD) approaches – or even some mixed or heuristic approaches. No matter in which form, an algorithm can always be defined in three aspects: Select – Serve – Evaluate, among which the critical aspect is "Select". Given a set of storage totes, how to "select" the storage locations in order to maximize the system's throughput? For a set of retrieval tasks, which sequence should be selected by the retrieval lift so that the makespan is minimized? For a particular shuttle, how should it select the next task

considering that both task types exist in its queue? When a tier-transfer service is required, which shuttle is the best candidate to be selected? Generally speaking, these decisions are made based on the knowledge of the system's current state, and to some extent the forecast of the system's future states. After the current service is performed, the quality of this decision is evaluated, and both short-term effects and long-term effects to system performance are considered and updated as the basis for the upcoming decisions.

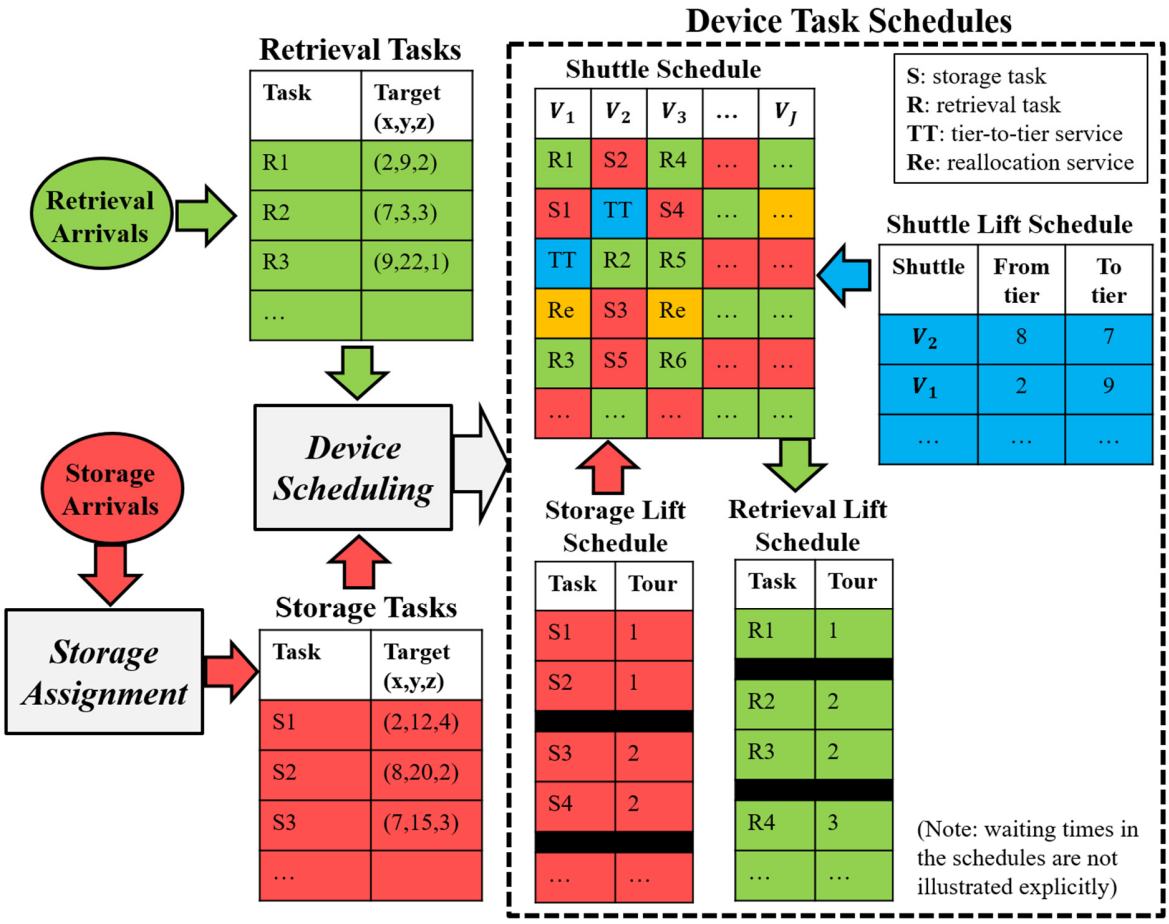

Figure 5.1 Operational Control Decisions in an SBS/RS aisle

As with the previous chapter, the focus of this chapter is on single SBS/RS aisles, thus higher-level decisions like material resource planning or global coordination are not discussed here. As illustrated in Figure 5.1, the operational control decisions under study in this chapter are categorized into two types:

Storage Assignment decisions, which create storage tasks by specifying the target storage location (slot) for each tote to be stored. It is assumed that the inbound totes are delivered to the aisle by a single conveying mechanism in First-Come-First-Serve (FCFS) patterns (which is the most common case). The storage assignment decision is made when the tote is loaded to the storage lift (except in dedicated storage policy). In addition, with 2-deep racks, the new storage location for the blocker tote to be relocated also need to be assigned if the target slot of the retrieval task is blocked.

Device Scheduling decisions, which determine the operation sequences for all four types of devices in order to complete the tasks. The device scheduling process takes tasks created from slotting and dispatching decisions as inputs and make the following decisions:

a) The sequence that each shuttle serves storage and retrieval tasks on its current tier;

b) The sequence and tours that the retrieval lift serves retrieval tasks;

c) The sequence and tours that the storage lift serves storage tasks; and

d) The sequence that the shuttle lift provides tier-transfer services and the shuttle to be selected in each service.

As illustrated in Figure 5.1, the control decisions are highly dependent on each other. Theoretically – given a known set of retrieval and storage tasks, and assuming the device service times are deterministic and there are no device failures – there exists at least one optimal solution that creates and completes tasks for a given objective (e.g., makespan or total tardiness). In such cases, the two types of control decisions are made simultaneously in a single MP form algorithm. However, different from the batching approaches widely applied in single-crane AS/RSs, with a given task set the start times and end times of the devices will be different. Because the task set is always dynamic as tasks continuously arrive to the aisle, it can be expected that such all-in-one approaches will not ensure the overall optimality for SBS/RSs. Moreover, concerning the complexity in device interactions, the difficulty of combining the storage assignment problem and the scheduling problem, as well as the stochasticity from the demands, the “all-in-one” approach is mathematically impractical in improving the system’s sustainable performance. A good control strategy is expected to divide-and-conquer those control problems and provide the system with effective, efficient, robust, and sustainable control solutions.

Before further introducing our approaches for developing the operational control strategies, it is necessary to clarify at first the differences between the assumptions made here and the assumptions made in the previous chapter when developing the queuing-based analytical approach. In the previous chapter, both the storage assignment policies and the device scheduling policies are simplified based on a set of assumptions. The storage assignment policies were assumed as random with equal probabilities between tiers and assumed as either Pure-Random-Storage or Closest-Open-Location for slots within tiers. Also, the relocation services for retrieval tasks were assumed as consistent with the storage assignment policies. For the device scheduling policies, the tote lifts were assumed to serve the tasks in FCFS patterns and try to load as many totes as possible to their current tour (if capacity > 1). The shuttles were assumed to serve the tasks in FCFS, at the same time perform dual-cycles as much as possible. With tier-to-tier configurations, tier-transfer tasks were assumed to be served in FCFS, and the shuttle to be transferred in each task was assumed to be selected randomly from all idle shuttles. SKU-level characteristics were not considered in the last chapter – the task arrivals were only described by the means and variances of their inter-arrival times, and each task was assumed as corresponding to a unique tote. Finally, the capacities of the input buffer and the output buffer on each tier were both assumed infinite.

The simplification assumptions mentioned above made it possible (although still difficult) to establish a generalized travel-time model for the systems. The estimation results obtained from that analytical approach are viewed as reasonable baselines of the systems' performances, and thus valid and efficient for evaluating large numbers of designs/configurations during the early design conceptualization (screening) phase. In this chapter, the focus is the later operational control phase, in which the objectives are further exploration of the systems' potentials and optimization of the systems' performance by customizing the control strategy according to the actual operational environments. In this chapter, the control strategy focuses on the SBS/RS operations and does not include the coordination with the upstream/downstream systems. Thus, the previous assumptions are relaxed to present the practical situations more accurately. The assumptions for the control strategy development are updated as follows:

1. Each tote contains a single SKU-type. Each task, either storage or retrieval, corresponds to one specific tote. The probability that a storage task stores a particular SKU-type, as

well as the probability that a tote of a particular SKU-type is requested by a retrieval task, are both subject to known distributions based on the SKU-level characteristics.

2. Storage assignment decisions and relocation decisions need to be specified for each storage tote / blocker tote based on the SKU-type and various indicators of the systems' current states.
3. Schedules for storage lifts are always FCFS due to the nature of this process, but the tours need to be specified.
4. Schedules for retrieval lifts need to be determined based on various indicators of the systems' current states, and the tours need to be specified.
5. Schedules for each shuttle, which may include both storage tasks and retrieval tasks, need to be determined based on various indicators of the systems' current states.
6. For tier-transfer services, both the shuttle to be transferred and the target tier need to be determined based on various indicators of the systems' current states.
7. I/O buffers have limited capacities: a storage lift will be suspended if the input buffer of the target tier of the current task is full; a shuttle will be suspended if the output buffer of the tier of the current task is full.

5.3 Formulation of the Control Problems

5.3.1 Device Scheduling

At a high level, the device scheduling approaches in SBS/RS are dealing with a scheduling problem with heterogeneous resources. Figure 5.2 illustrates the task schedule patterns for each device type: assuming four storage tasks (S1 to S4) and four retrieval tasks (R1 to R4) simultaneously arrive to an aisle deployed with two shuttles, and the initial states of all devices are idle. There are various precedence constraints in this scheduling problem. First of all, the shuttles interact with tote lift services through I/O buffers positioned at the front side of each tier, and the downstream services (shuttle services for storage tasks, lift services for retrieval tasks) cannot be started prior to the completion of the upstream services (lift services for storage tasks, shuttle services for retrieval tasks) – as indicated by the shadowed areas in the above schedule charts. Secondly, because those I/O buffers are usually roller conveyors on which tote sequences cannot be altered, the sequences of the downstream services are also constrained by those of the upstream services. Thirdly, the touring characteristics the lift services further increase the complexity of the

problem – in this example, both tote lifts have tour capacity of two, and each tote lift has performed two full-size tours (thus the service times of the tasks in the same tour overlap). Moreover, the planning of tier-transfer operations adds another subset of constraints – in this example, after Shuttle.1 completed task S1, it is transferred to another tier to proceed with task R2. The shuttle lift is occupied then and no other tier-transfer operation can be processed until Shuttle.1 is transported to the target tier. At last, all the constraints discussed above were assumed independent from the storage assignment problem (including the relocation decisions). If integrated with the storage assignment decision for each task, it will be impractical to obtain mathematical optimality for the scheduling problem even for small systems and moderate sizes of task sets (e.g. 24 tasks for a 12-tier, 6-shuttle rack).

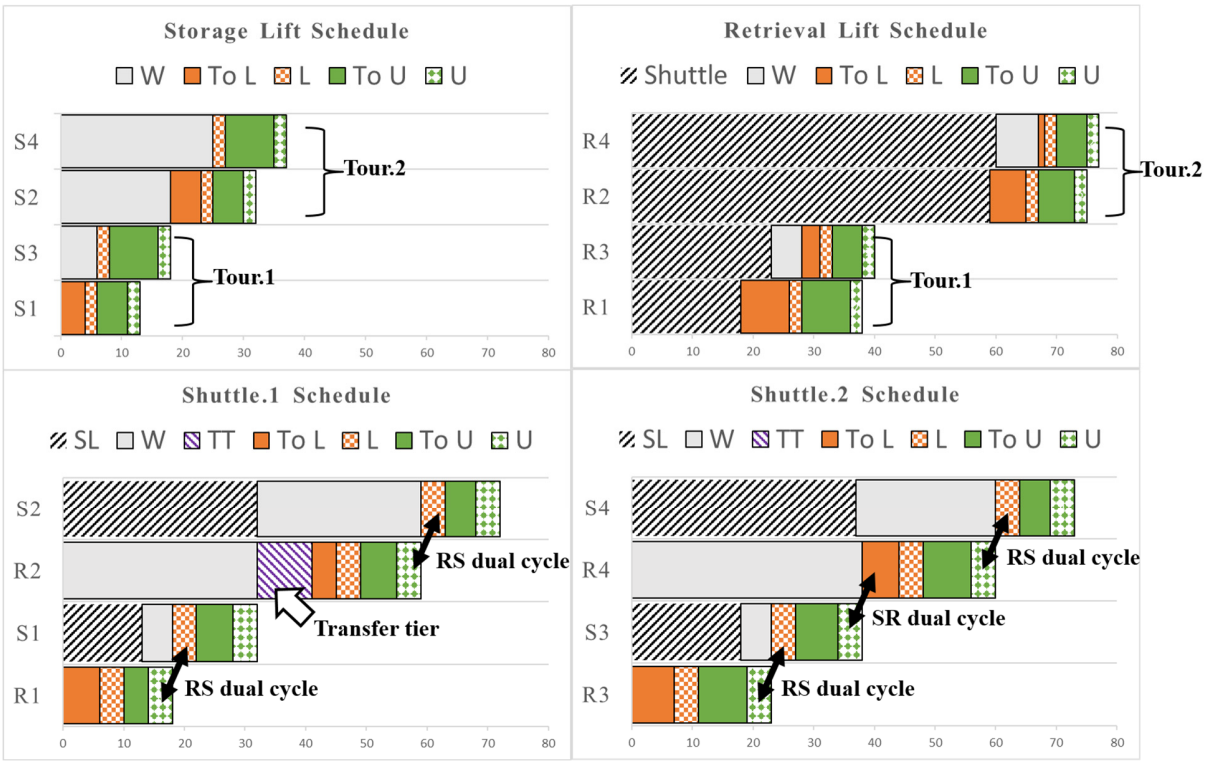

Figure 5.2 Task Schedules of SBS/RS device (W: waiting device response; To L: moving to load point; L: load; To U: transporting tote to unload point; U: unload; TT: tier-transfer operation; SL/Shuttle: in previous service stage)

5.3.2 Storage Assignment

The storage assignment problems in various types of warehousing systems are based on a common ground that the costs (primarily travel times) for accessing different storage locations are different. In traditional AS/RSs with one crane in each aisle, the storage locations are visited by a single crane, the tasks are performed sequentially, and the distances of the tasks' target slots from the rack's I/O point(s) are the primary factors that determine the crane's travel times. Because the crane is the only L/U device that accesses the storage locations (slots), the time costs for accessing each slot in the rack can be presented in relatively simple mathematical expressions described by the slots' (x, y) coordinates and crane velocity and acceleration, etc. In many classic storage assignment approaches, differences in such accessibilities and differences in SKU-level characteristics are considered together to minimize the overall travel costs – for example, SKUs with higher demands or/and turnover rates are usually stored to locations easier to access (e.g., closer to I/O).

In an SBS/RS aisle, the time costs for accessing the slots are more difficult to evaluate due to the complexity of the service patterns. Tasks are served in sequence by the tote lifts but served in parallel by the shuttles – the vertical and horizontal service stages are more decoupled than the diagonal service patterns in traditional AS/RSs, but still to some extent coupled due to factors like the buffer characteristics as discussed earlier this section. Figure 5.3 presents a general map to illustrate the slot accessibilities within an aisle. Each slot location in the aisle can be described as three-dimension coordinates (x, y, z), where x is the tier index, y is the column index and z is the depth index. The time costs to access a slot (x, y, z) is determined by both the tote lift travel times from their I/O terminals (function of x) and the shuttle travel times from the I/O buffers on the tiers (function of y) – it is noticeable that these two factors are not necessarily evaluated with the same importance. For example, if the shuttle utilizations are much larger than the tote lift utilizations in a particular system configuration under a particular demand scenario, then it is probably more important to assign totes to slots more accessible for the shuttles rather than for the tote lifts (thus more weight regarding y and less weight regarding x). With 2-deep racks, the accessibility of each 2-deep slot ($z \in [3,4]$) needs to include not only the longer load/unload times, but also the occurrence of relocation operations and the corresponding time costs – as discussed in Chapter 4 the relocation factor is usually much more significant to task cycle times. With tier-to-

tier configurations, the time costs for accessing the slot are also subject to the number of shuttles deployed as well as the occurrence of tier-transfer operations.

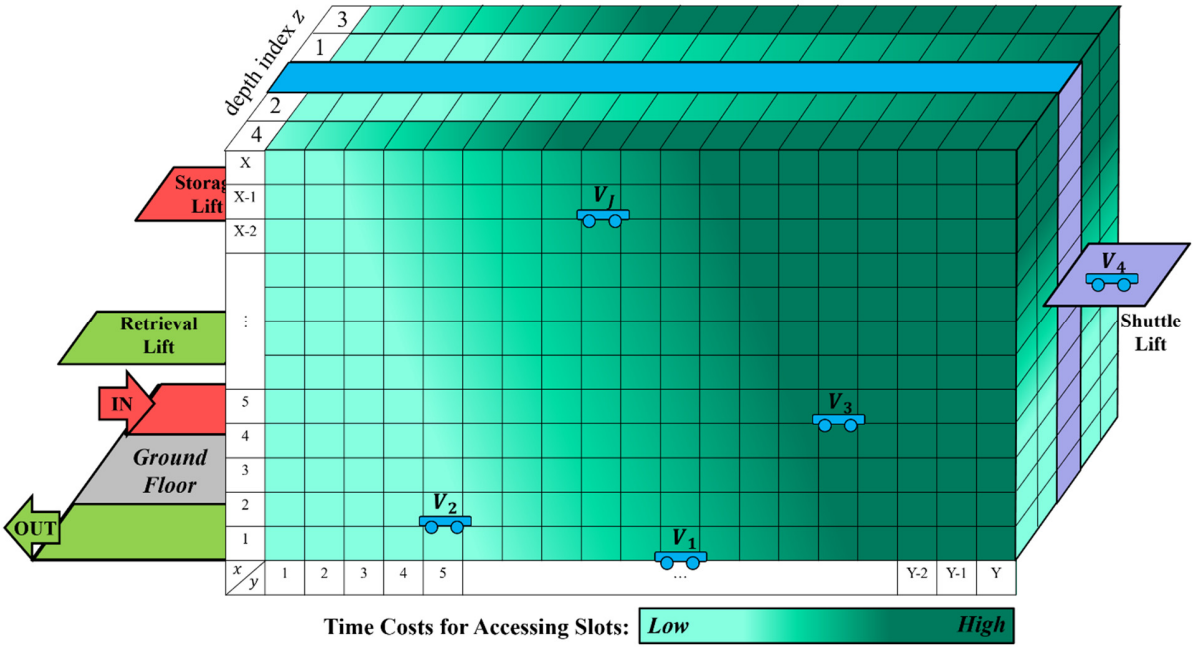

Figure 5.3 Illustration of time costs for accessing rack slots in a 2-deep SBS/RS aisle

5.3.3 Insights obtained from the Travel Time Model

As discussed above, multiple decisions need to be made for different task types, different device types, by the control strategies. First, based on the complexities of the system discussed previously, it is impractical to develop an effective and economic all-in-one mathematical programming (MP) approach that optimizes the sustainable performance of the systems. Secondly, it is impractical to explore and evaluate all the possible combinations of all types of dynamic dispatching (DD) approaches, as well as proving their effectiveness for various system designs and system configurations under different demand scenarios. However, from the analytical approach introduced in Chapter 4 many important insights are obtained from the processes of the travel time model development and from the validation experiment results, summarized as followed:

1. Both the utilizations of the shuttles (U^V) and the utilizations of the tote lifts (U^{SL} and U^{RL}) could form bottlenecks on the systems' throughput. With the same rack capacities, U^{SL} and U^{RL} are typically higher for taller racks (larger X). Also, U^V is in general larger for deeper racks (larger Y), but the number of shuttles deployed (J) is a more significant factor to it. Thus, the control approaches should be designed to accommodate to different system designs and configurations.
2. With the same demand scenarios, system throughput is primarily determined by the average service times of the devices T^V, T^{SL} and T^{RL}, in which T^V is generally the most significant especially for deeper racks (larger Y). Good control approaches, including storage assignment and device scheduling, can reduce the average service times and thus improve the system's throughput performance.
3. In order to reduce the service times, storage assignment approaches should aim at increasing the task visit probabilities of the slots ($P^S, P^R, P^{RelI}, P^{ReO}$) that are easier to access (e.g., slots closer to the I/O buffers) and decreasing the probabilities of the slots that are harder to access. Moreover, with the presence of relocation services (2-deep racks), minimizing the occurrence of relocation operations (θ^R) is expected to be beneficial. Finally, knowledge of SKU-level characteristics should be incorporated in the storage assignment approaches, if possible (e.g., SKUs with higher demands should be stored to slots that are easier to access).
4. In order to reduce the service times, because the load/unload times are deterministic, device scheduling approaches should aim at reducing the travel times in serving the tasks. For the shuttles, this resembles a Traveling Salesman Problem (TSP) in which each shuttle task is viewed as a node to be visited, and the travel costs between such nodes are deterministic, and larger portion of dual-cycles (θ^D) is expected to be beneficial. For the retrieval lifts, it is speculated beneficial to schedule the tasks in a way that minimizes the total travel time of each tour. For the storage lifts, there is no scheduling flexibility because the storage arrivals are FCFS, but the tour times are affected by the storage assignment approaches.
5. A shuttle cannot start serving a storage task prior to the end of storage lift service of that task; likewise, a retrieval lift cannot start serving a retrieval lift prior to the end of shuttle service of that task. Such coupling effects between the shuttle services and the tote lift services need to be considered in the scheduling approaches to minimize the intermediate task waiting times on the I/O buffers, and the buffer capacities need to be concerned.

6. With tier-to-tier configurations, the occurrence of tier-transfer tasks (θ^T) should be minimized to reduce the overall shuttle service times. The scheduling decisions for tier-transfer services are two-stage, in which both the target tier and the service shuttle need to be determined. The storage assignment decisions should also avoid introducing unnecessary tier-transfer workloads.
7. Given the system configuration and demand scenario, the task waiting times are largely affected by the coefficient of variations of task inter-arrival times (cov_a) and task service times (cov_s) for each device. Good control approaches are expected to reduce such variances in the long term and improve the system's responsiveness performance.

5.4 Exploration of Scheduling Algorithms based on MP Approaches

As discussed in Section 5.2, there are generally two types of operational control options to improve the performances of an SBS/RS aisle:

1) **Mathematical Programming (MP) Algorithms** (*Top-down approaches*)

According to the definitions by Chandra and Grabis (2007), the general purpose of mathematical programming is finding an optimal solution for allocation of limited resources to perform competing activities. Mathematical programming uses a compact mathematical model for describing the problem of concern. The solution is searched among all feasible alternatives. In our context, the term "solution" here includes both the task schedules of all devices and the target slots of the storage assignment decisions.

2) **Dynamic Dispatching (DD) Algorithms** (*Bottom-up approaches*)

In our context, dynamic dispatching algorithms refer to the approaches in which decisions are made dynamically and step by step. Instead of searching for optimal solutions, the focus here is on providing "good-enough" solutions that ensure robust and sustainable system performance.

As discussed before, MP approaches alone are not practical here considering the complexity inherent in the system service patterns and the dynamic and stochastic nature of demands. The DD approaches appear to be more feasible and easier to formulate. However, considering the large number of potential inputs for the dynamic dispatching algorithms (device

status, task progresses, slots occupancies, SKU characteristics…), it is impossible to explore and evaluate all the possible combinations of all types of the DD approaches.

The operational control strategy development approach proposed in this research integrates explorations of both MP algorithms and DD algorithms. Based on the observations and insights obtained from both the analytical approaches and simulation approaches, the following assumptions are made: the DD algorithms need to be developed systematically and guided with some baselines or/and benchmarks to evaluate their qualities and to narrow down the search scopes for improving the algorithms – and such baselines/benchmarks are expected to be provided by MP approaches.

In this section, a divide-and-conquer methodology is applied, and the problem scope is expanded gradually. The problem scope is first narrowed to the scheduling of a single, tier-captive shuttle – which is the most rudimental form of the system – based on a set of simplifying assumptions. The MP formulation is developed for that basic system. The complexity of the MP formulation is gradually increased along with the increasing of the problem scope and the elimination of the previous simplifying assumptions – the MP solution techniques applied here involve both classic ones like Integer Programming and heuristic ones like Ant-Colony-Optimization – until a point that the MP is no longer practical to form or solve. On the other hand, the DD algorithms are developed, evaluated, and fine-tuned by comparing the DD solutions with the MP solutions along the entire scope-expansion process. This process is fully simulation-based, and the solution qualities of both MP and DD are examined under various system designs/configurations and demand scenarios.

5.4.1 Shuttle scheduling in tier-captive SBS/RS

In this subsection we focus on the scheduling problem of a single shuttle in a tier-captive system. The coupling affects from the tote lifts, the tier-transfer operations, and advanced storage assignment approaches are not included in the current scope. As illustrated in Figure 5.4, four problem formulations are identified for the tier-captive shuttle scheduling problem according to task complexity and control assumptions. Problem 1a is the most basic formulation and uses a classic Traveling Salesman Problem (TSP) which is solvable through Integer Programming (IP) approaches. Problem 1b and 1c are more complicated formulations developed based on Problem 1a, in which precedence constraints between tasks are added according to assumptions for storage

assignment and relocation, respectively – these problems already become much more costly for IP approaches compared to the basic problem. Problem 1d is the most complicated formulation in the current context which integrates the mathematical complexities from Problem 1b and 1c. Note that in this dissertation, our focus is on exploring the complexity of the scheduling problems rather than on solving these problems.

Problem 1a: Scheduling of a single, tier-captive shuttle with indistinctive storage assignment and no relocation

This Problem 1a is identified as the baseline for the Problems 1b, 1c and 1d to be introduced later in this section. The problems' scopes are limited to a single shuttle on a single rack tier. Given the target slots of the storage tasks and the retrieval tasks, each problem identified in this section here can all be presented as Traveling Salesman Problem (TSP) where each task is viewed as a node to be visited. Precedence constraints may exist between nodes. For example, from the perspective of the totes to be stored, due to the roller-conveyor characteristics of the input buffer, their service sequence by the shuttle cannot be altered from the original sequence when they entered the input buffer, thus introducing precedence between storage totes.

However, in this Problem 1a the storage totes are assumed indistinctive, which means all empty slots in the rack tier are equally available to each incoming storage tote, and the tote's target slot is determined when the shuttle starts serving the storage task instead of predetermined at the beginning. With this assumption, the problem becomes a classic TSP and the storage precedencies are removed from the formulation. This assumption is solid when relatively simple storage assignment rules are applied to the system, e.g., Pure Random Storage (PRS) or Closest-Open-Location (COL) regardless of the SKU-level characteristics – in such cases, a set of available slots is assumed selected according to the storage assignment rule, and the size of this set equals the number of totes to be stored. Moreover, blocker-relocation for retrieval tasks is not considered in this baseline problem, which means the target slot of each retrieval task can be accessed without extra relocation efforts.

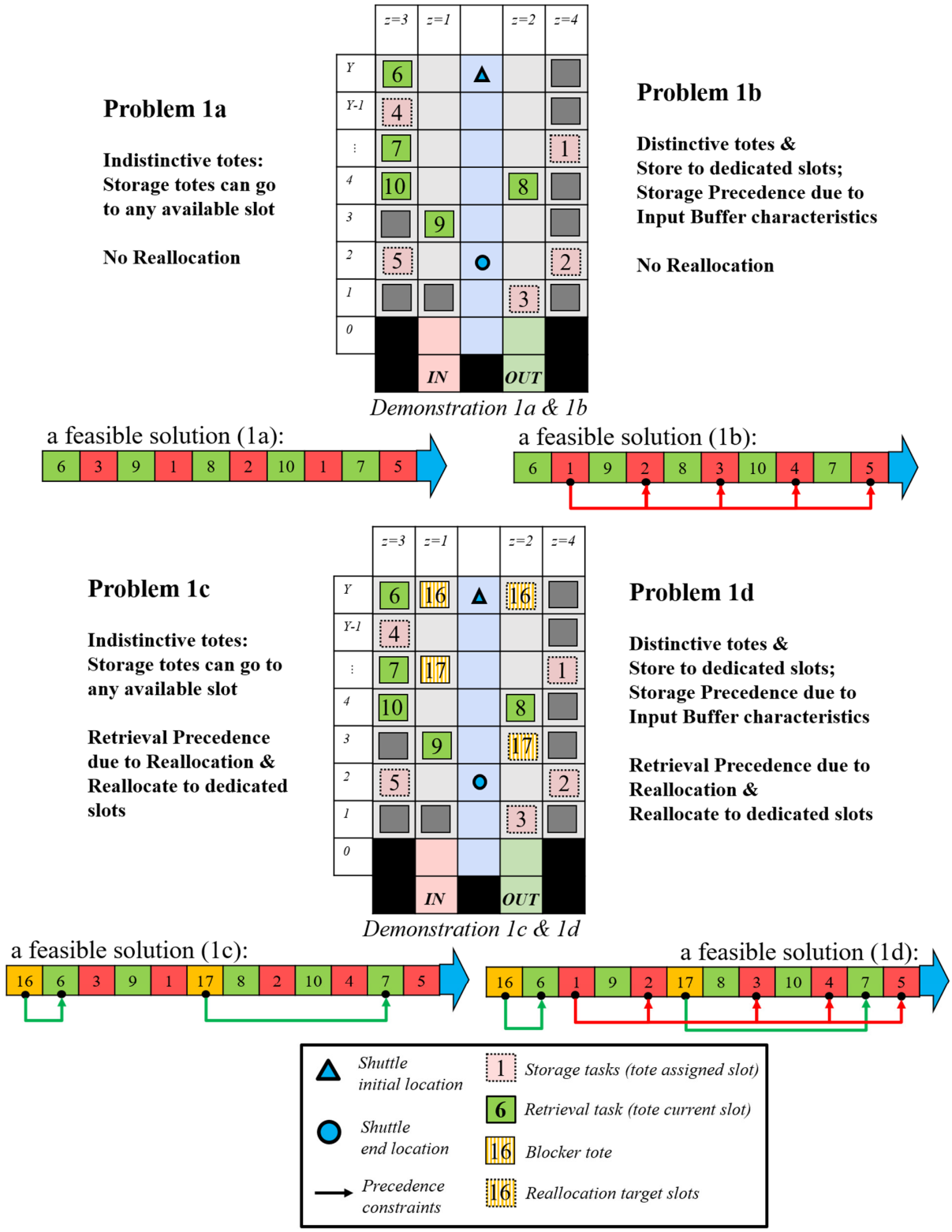

Figure 5.4 Problem formulations of shuttle scheduling in Tier-captive SBS/RS (1a, 1b, 1c, 1d)

Problem Inputs:

S : Number of storage tasks;

R : Number of retrieval tasks;

$\boldsymbol{A_S}$: A set of slot locations selected for the S storage tasks according to some given storage assignment policy, where the size of the set $|\boldsymbol{A_S}| = S$. It is assumed that the storage totes are indistinctive thus each tote can be assigned to any slot $[x_i, y_i, z_i] \in \boldsymbol{A_S}$.

i : Task index, where $i \in \{0, 1 \cdots (S+R)\}$. The subset $\{1,2 \cdots S\}$ are for storage tasks and the subset $\{(S+1), (S+2) \cdots (S+R)\}$ are for retrieval tasks. In addition, an index $i = 0$ is a dummy task that indicates the starting point of the schedule. In the following steps, for the simplicity of formulation, the task type π_i is used to indicate the subset that task i belongs to, where:

$$\begin{cases} \pi_i = 1 & if\ i \in \{1, 2 \cdots S\} \quad (Storage\ tasks) \\ \pi_i = 2 & if\ i \in \{S+1, S+2 \cdots S+R\} \quad (Retrieval\ tasks) \end{cases}$$

$[x_i, y_i, z_i]$: Target slot location of task i – for a retrieval task, the target slot is the slot from where the tote is to be retrieved; for a storage task, the target slot is the slot where the tote is to be stored, and $[x_i, y_i, z_i] \in \boldsymbol{A_S}$ for $\forall i \in [1, 2 \ldots S]$. According to the problem definition, all tasks are assumed to be at the same tier, thus $x_1 = x_2 = \cdots = x_{S+R}$. For simplicity of problem formulation, it is assumed that in this problem that the target slots are all different, thus $y_{i_1} \neq y_{i_2}\ or\ z_{i_1} \neq z_{i_2}$ for $\forall i \in [1, 2 \ldots S+R]$ (which means for the given task set, a storage tote cannot be assigned to the target slot of a retrieval task even if this slot became empty after the retrieval). It is further assumed that if the rack is 2-deep, all slots $\in \boldsymbol{A_S}$ are not neighboring slots in the same column, thus $y_{i_1} \neq y_{i_2}\ or\ z_{i_1} \neq (z_{i_2} + 2)$ for $\forall i \in [1, 2 \ldots S]$ (otherwise the storage tasks will be subject to precedence constraints). Finally, it is assumed in this problem that all retrieval tasks will not need relocation (thus the totes to be retrieved are either stored in 1-deep slots or stored in non-blocked 2-deep slots).

d_i : The remaining service time of task i from the time the shuttle arrived its load point, computed as:

$d_0 = 0$, because the initial state of the shuttle is assumed idle;

$$d_i = \tau^V_{0,\,y_i} + \omega^V_0 + \omega^V_{z_i} \quad \forall\, i \in [1, 2 \ldots (S+R)]$$

Where $\tau^V_{0,\,y_i}$ is the shuttle travel time between I/O location and the target column y_i, ω^V_0 is the shuttle L/U time from/to input/output buffers, and ω^V_z is the shuttle L/U time from/to slot of depth index z. It can be inferred that in any feasible schedule, d_i's are the same. Denote D as the *fixed costs* of the scheduling problem, where:

$$D = \sum_{i=0}^{S+R} d_i$$

y_0 : The initial column location of the shuttle when it becomes idle to the task set – in this problem, the shuttle is assumed idle at the beginning.

$c_{i_1,\,i_2}$: If task i_2 is served after task i_1, the total time costs from the start (load point) of task i_1 to the start location (load point) of task i_2 (incl. service time of i_1), computed as:

$If\ i_1 = 0\ (thus\ i_2\ is\ the\ first\ task\ in\ schedule)$:

$$c_{i_1,\,i_2} = c_{0,\,i_2} = \begin{cases} \tau^V_{y_0,\,0} & if\ i \in \{1 \cdots S\}\ (Storages) \\ \tau^V_{y_0,\,y_{i_2}} & if\ i \in \{S+1 \cdots S+R\}\ (Retrievals) \end{cases}$$

$Else\ If\ i_2 = 0\ (thus\ i_1\ is\ the\ last\ task\ in\ schedule)$:

$$c_{i_1,\,i_2} = c_{i_1,\,0} = d_{i_1}$$

$Otherwise$:

$$c_{i_1,\,i_2} = \begin{cases} d_{i_1} & if\ \pi_{i_1} = 2, \pi_{i_2} = 1 \quad (R - S\ case) \\ d_{i_1} + \tau^V_{y_{i_1},\,y_{i_2}} & if\ \pi_{i_1} = 1, \pi_{i_2} = 2 \quad (S - R\ case) \\ d_{i_1} + \tau^V_{y_{i_1},\,0} & if\ \pi_{i_1} = 1, \pi_{i_2} = 1 \quad (S - S\ case) \\ d_{i_1} + \tau^V_{0,\,y_{i_2}} & if\ \pi_{i_1} = 2, \pi_{i_2} = 2 \quad (R - R\ case) \end{cases}$$

Where $\tau^V_{y_1,\,y_2}$ is the shuttle travel time between column y_1 and y_2, and obviously $\tau^V_{y_1,\,y_2} = \tau^V_{y_2,\,y_1}$;

Task arrival time assumptions:

All retrieval tasks are assumed available for shuttle service from time 0. Also, there is assumed always one storage task already arrived the exit point of the input buffer and thus available

for shuttle service (which means the storage lift never delays the shuttle's schedule). Together with the previous assumption that all S storage totes are indistinctive in storage assignment to the S storage locations, the shuttle scheduling problem is simplified into a classic Traveling Salesman Problem (TSP) with $(S + R + 1)$ nodes: each task is viewed as a node to be visited where the node is the load location of the task, and the distance between each pair of task nodes is the service time of the previous task since the shuttle arrived its load point, plus the travel time from the unload location of the previous task to the load location of the current task. It is noticeable that the costs between nodes are asymmetric.

Decision Variables:

x_{i_1, i_2} : Binary variable (BV) indicating if task i_2 immediately follows task i_1 in the shuttle's schedule, where $i_1, i_2 \in [0, 1, 2 \dots S + R]$.

(Note: the character x in x_{i_1, i_2} here is the notation for BVs in Integer Programming (IP), which is distinguished from the tier index x for indicating in slot location $[x, y, z]$ – the BVs are noted with x – to be consistent with traditional TSP-IP formulation. These two notations will not appear at the same time within any formula in this research.)

u_i : Integer variables for excluding sub-tours based on Miller–Tucker–Zemlin (MTZ) formulation, where $i \in [0, 1, \dots S, (S + 1) \dots (S + R)]$.

Objective Function:

In this problem, the focus is only on scheduling of a single, in-tier shuttle, and the storage locations of all storage tasks are assumed determined at the beginning. Thus, the shuttle's schedule and the corresponding objective function are only determined by the decision variables x_{i_1, i_2}. The total makespan of all tasks is determined as the objective to be minimized: according to the problem definitions, the shuttle is always busy until all tasks are completed, thus minimizing the total makespan is equivalent to maximizing the average task service rate of the shuttle. Hence, the objective function is presented as:

$MIN\ \ F(x)$, where:

$$F(x) = \sum_{i_1=0}^{S+R} \sum_{\substack{i_2=0, \\ i_2 \neq i_1}}^{S+R} \left(c_{i_1, i_2} \times x_{i_1, i_2} \right) \quad (1)$$

The above formulation complies with the classic Integer Programming (IP) form for solving TSP. As discussed before, the lower bound of this problem is the fixed costs $D = \sum_{i=1}^{S+R} d_i$.

Constraints:

The constraints of this basic single-shuttle scheduling problem mostly comply with the constraint forms in the classic IP-TSP problems. First of all, the binary variables x_{i_1, i_2} must subject to the following constraints to ensure the schedule is valid and each task is only visited once:

$$\sum_{i_2=0}^{S+R} x_{i_1, i_2} = 1 \quad \forall i_1 \quad (2)$$

$$\sum_{i_1=0}^{S+R} x_{i_1, i_2} = 1 \quad \forall i_2 \quad (3)$$

$$0 \leq x_{i_1, i_2} \leq 1, x_{i_1, i_2}\ integer \quad \forall i_1, i_2 \quad (4)$$

In addition, the following constraints are formulated to exclude sub-tours from the schedule:

$$u_0 = 1 \quad (5)$$

$$2 \leq u_i \leq S + R + 1 \quad \forall i > 0 \quad (6)$$

$$u_{i_1} - u_{i_2} + 1 \leq (S + R)\left(1 - x_{i_1, i_2}\right) \quad \forall i_1, i_2 > 0 \quad (7)$$

The integer constraints and sub-tour exclusion constraints above together with the previous objective function formulation construct a Miller–Tucker–Zemlin (MTZ) formulation of TSP, where the last inequality is the arc-constraint.

It is noticeable that the above MTZ formulation still does not ensure optimality of the schedule, because the target slots of storages and retrievals are assumed not overlapping in this problem – consider the situation that a storage tote could be assigned to a slot from which a retrieval is made earlier in the schedule. In such situations, the problem is expected to be complicated by additional precedence constraints: some storages must wait for the completion of some retrievals so that the slots of the latter ones become available. However, we will not explore

such situations considering the fact that the number of available slots is usually much larger than the number of waiting tasks on an SBS/RS tier.

Evaluation:

In the above MTZ-TSP formulation, $(S+R+1)^2$ binary variables plus $(S+R+1)$ integer variables are defined, and totally $(S+R+1)^2+2(S+R+1)$ constraint rows are formulated to find the optimal solution. Given a task set with 10 storages and 10 retrievals, that will be 462 decision variables and 483 constraint rows – with this problem size, it takes roughly one second to solve the problem on an average 2.60GHz clock speed and 8GB RAM. The optimization problems formulated are solved using the SCIP Optimization Suit for Mixed Integer Programming (The SCIP Optimization Suite 8.0, 2021). This optimization approach may become costly with large problems, because the numbers of decision variables and the numbers of constraint rows both increase exponentially to the sizes of the task sets. Moreover, the problem discussed here is just a simplified problem formulated for a single shuttle, and the interactions with other devices as well as relocations are not taken into consideration yet. In addition, the stochastic characteristics of task arrivals require the optimization algorithm to be called iteratively whenever the contents in the task set is changed – not to mention that the "optimal solution" found in each iteration could be much less meaningful for the system's long-term, sustainable, and robust performance. Considering the timing requirements for device scheduling decisions of SBS/RS are typically within a few hundred milliseconds per task, the MP approach like this problem appear to be less practical when either the problem size or the problem complexity increases.

However, important observations can be made based on the optimization formulation to guide the development of more practical control algorithms – which is the goal of all the efforts of exploring MP approaches in this chapter. As observed in the formulation, the lower bound of the scheduling solution is a fixed cost $D=\sum_{i=1}^{S+R} d_i$. Also, considering the rectilinear travel pattern of the shuttle within a tier, the solution space of the shuttle scheduling problem formulated here is expected to be relatively smooth – which means, near-optimal solutions may be obtained through some much less costly and more sustainable scheduling approaches like Dynamic Dispatching (DD). For example, the Dual-Cycle approach introduced in Chapter 4 is one of the typical DD approaches based on the shuttles' service patterns and increasing the DC occurrences could

potentially improve both the throughput performance and the responsiveness performance of the system. To evaluate the potential of DD approaches, three simple DD rules are tested:

1) FCFS-DC Policy: the shuttle performs tasks in Dual Cycles and in first come-first serve pattern as the scheduling assumptions made in Chapter 4. Once a task is completed, the next task to be served is the first task of the different type in the remaining tasks – if no different type task exists, then the next task to be served is the first task of the same type. In addition, the first task to be served is either the first storage task ($i = 1$) or the first retrieval task ($i = S + 1$) whichever the travel time from the shuttle's initial location y_0 is smaller. For example, with two storage tasks ($i \in \{1,2\}$) and retrieval tasks ($i \in \{3,4,5\}$) (assuming the indices of both task types are consistent with task arrival times, respectively) and $y_0 = 0$ (thus the shuttle is at I/O), then task 1 (storage) will be selected as the first task, and the resulting schedule will be $[1 \to 3 \to 2 \to 4 \to 5]$.
2) Closest-First Policy: the shuttle selects the next task whichever the travel time from the shuttle's current location is smaller among all the remaining tasks. The first task to be served is the one with the smallest $c_{0,i}$ from the shuttle's initial location. Once a task i_1 is completed, the next task i_2 to be served is the one with the smallest c_{i_1,i_2}, where $i_2 \in \{Remaning\ Tasks\}$.
3) Dijkstra's Algorithm: which is a greedy algorithm for finding the shortest path to visit a set of nodes. In our implementation of the algorithm for the current problem formulation, the shuttle selects each task in the schedule based on two components: firstly, the travel time from the shuttle's current location to the load point of the candidate task; and secondly, the travel time from the unload point of the candidate task to the closest load point among all remaining tasks. Thus, once a task i_1 is completed, the shuttle selects the next task i_2 whichever has the smallest tentative distance value $f(i_2) = c_{i_1,i_2} + \min(c_{i_2,i_3})$, where $i_2, i_3 \in \{Remaning\ Tasks\}$ and $i_3 \neq i_2$.

Based on a single tier of 200 columns, 40 task sets of different sizes are created randomly to test the MP approach (optimization) as well as the DD approaches. The task sets include 10 random sets for each of the four cases $(S = 5, R = 5), (S = 5, R = 10), (S = 10, R = 5), (S = 10, R = 10)$, thus the sizes of storages and retrievals are not necessarily equal. The target slots of the tasks are selected randomly with equal probabilities. To present the typical shuttle service pattern as discussed in Chapter 4 the shuttle's initial location is either the I/O column ($y_0 = 0$)

with 50% probability, or one of the rack columns ($y_0 = y \in [1,2 \dots Y]$) with equal probability of $50\%/Y$ for each column. The system parameters including rack dimensions, shuttle velocity/acceleration and load/unload times are the same settings as in 4.6.1. The optimization and DD results (makespan of the task sets) are obtained for each task set, and the result averages of the 40 task sets are illustrated in Figure 5.5. In addition, the average fixed costs of the task sets are displayed as baselines of the scheduling results, and the average results of full random scheduling are presented as references.

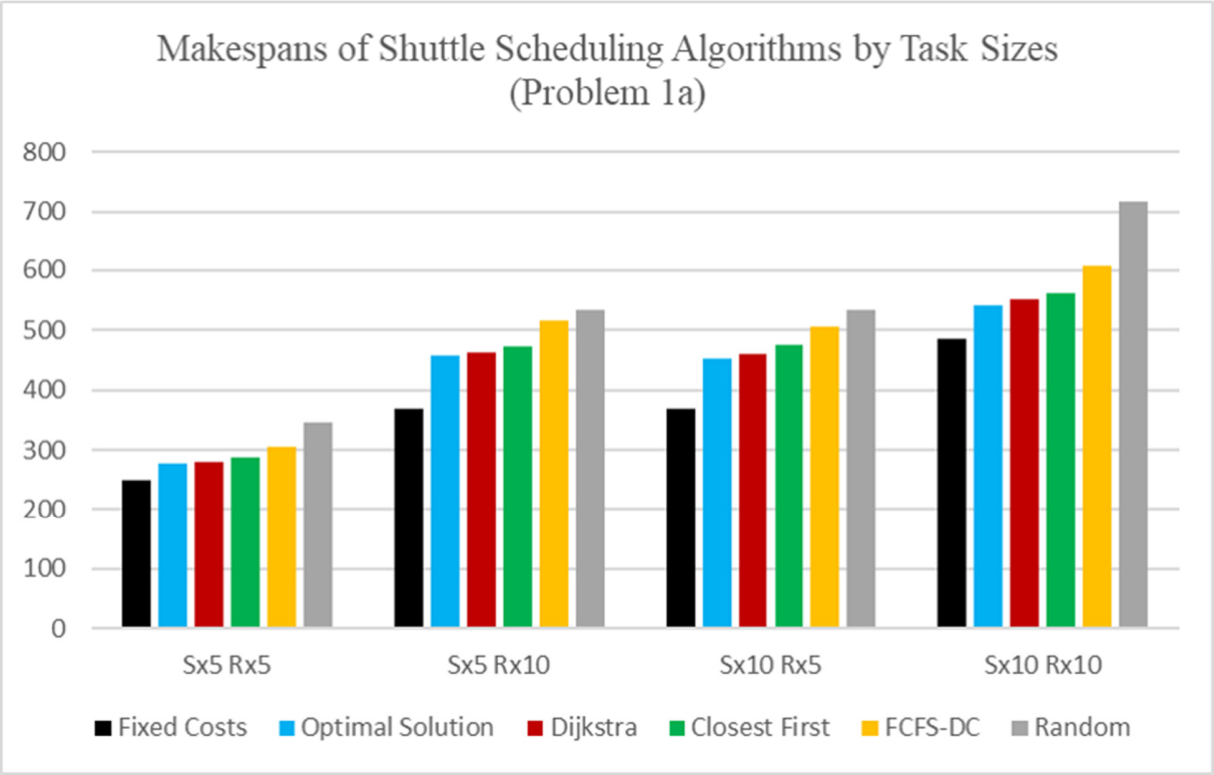

Figure 5.5 Dynamic Dispatching Results vs. Optimal Solution (Problem 1a, algorithm averages of 10 problems for each task size)

In Figure 5.5, it can be observed that the results from the Dijkstra's Algorithm are very close to the optimization results. As discussed before, the fixed costs make up the majority of the makespan. For the three DD approaches: Dijkstra's, Closest-First and FCFS-DC, their gaps to the optimization results are 1.3%, 3.9% and 11.6%, respectively. All three DD approaches appear to be effective compared with random scheduling, while both the Dijkstra's Algorithm and the Closest First policy further outperformed the FCFS-DC (which is the default scheduling policy in

the analytical approaches proposed in Chapter 4). It is noticeable that the results here do not indicate any direct conclusion for the control strategy development, because the problem formulated here is based on many simplifying assumptions that do not apply in practical system operations. Instead, both the Dijkstra's Algorithm and the Closest-First Policy are viewed as attractive candidates based on which the control strategy may be developed.

Problem 1b: Storage tasks with dedicated slots and precedence constraints

Regarding the storage assignment techniques applied in the system, more constraints may need to be considered in the MP formulations. Because the buffers are assumed to be roller-conveyors, the service sequence of storage tasks cannot be altered from the original sequence that the totes entered the input buffer. As a result, service precedencies exist between every two storage totes that arrive the input buffer one after another. In Problem 1a, because the storage totes were assumed indistinctive, such precedencies were neutralized by the flexibilities in choosing the slot for each storage tote and thus not need to be formulated as constraints. However, in situations where the slot is predetermined for each storage tote at the beginning, such precedencies need to be formulated to ensure feasible schedules.

For the simplicity of the formulation, it is again assumed that the storage slots and the retrieval slots do not overlap in the given task set (otherwise, additional precedencies need to be introduced between storage tasks and retrieval tasks).

Problem Inputs:

The same as in Problem 1a, except for that the storage totes are distinctive, thus each storage slot $[x_i, y_i, z_i] \in \boldsymbol{A_S}$ of all storage tasks $i \in \{1, 2 \cdots S\}$ is dedicated for the corresponding tote. Storage task i cannot be served by the shuttle before storage task $(i-1)$, $\forall i \in \{2, 3 \cdots S\}$ due to the roller-conveyor characteristics of the input buffer,

Decision Variables:

With the presence of the storage precedence identified above, we decide to apply the Two-commodity Network Flow Model purposed by Moon et al. (2002) for formulating TSP with precedence constraints. In this model, a commodity p is supplied by $(n-1)$ units at a selected starting node and consumed by one unit at each node that is not the starting node, and a

commodity q is consumed by $(n-1)$ units at a selected starting node and supplied by one unit at each node that is not the starting node – in our context, $n = (S+R+1)$, and the starting node is always the dummy task with $i = 0$.

With this approach, two additional sets of integer variables $x^p_{i_1,\,i_2}$ and $x^q_{i_1,\,i_2}$ are introduced to the problem formulation which will be explained later. Same as in Problem 1a, variables $x_{i_1,\,i_2}$ is an $(S+R+1) \times (S+R+1)$ matrix of binary variables indicating if task i_2 is the next task after task i_1 in the shuttle's schedule. The tour-feasibility variables u_i are not used here and the tour-feasibility constraints will be formulated with the new variables $x^p_{i_1,\,i_2}$ and $x^q_{i_1,\,i_2}$ instead.

Objective Function:

Assume the optimization objective is still to minimize the total makespan of all tasks (incl. relocations), with the Two-commodity Network Flow Model approach, the objective function is presented as:

$MIN\ \ F(x)$, where:

$$F(x) = \sum_{i_1=0}^{S+R} \sum_{\substack{i_2=0,\\ i_2 \neq i_1}}^{S+R} \left(c_{i_1,\,i_2} \times \frac{x^p_{i_1,\,i_2} + x^q_{i_1,\,i_2}}{S+R} \right) \tag{1}$$

Like in Problem 1a, the later part of the objective function is a constant because all d_i's are computed previously.

Constraints:

$$\sum_{i_2=0}^{S+R} x^p_{i_1,\,i_2} - \sum_{i_2=0}^{S+R} x^p_{i_2,\,i_1} = \begin{cases} (S+R) & if\ i_1 = 0 \\ -1 & otherwise \end{cases} \tag{2}$$

$$\sum_{i_2=0}^{S+R} x^q_{i_1,\,i_2} - \sum_{i_2=0}^{S+R} x^q_{i_2,\,i_1} = \begin{cases} -(S+R) & if\ i_1 = 0 \\ 1 & otherwise \end{cases} \tag{3}$$

$$\sum_{i_2=0}^{S+R} \left(x^p_{i_1,\,i_2} + x^q_{i_1,\,i_2} \right) = (S+R) \quad \forall i_1 \tag{4}$$

$$x^{p}_{i_1,\,i_2} + x^{q}_{i_1,\,i_2} = (S+R) \times x_{i_1,\,i_2} \quad \forall i_1, i_2 \tag{5}$$

$$\sum_{i_2=0}^{S+R} x^{p}_{i_1-1,\,i_2} - \sum_{i_2=0}^{S+R} x^{p}_{i_1,\,i_2} \geq 1 \quad \forall i_1 \in \{2, 3 \cdots S\} \tag{6}$$

$$x^{p}_{i_1,\,i_2} \geq 0, \;\; x^{p}_{i_1,\,i_2}\; integer \quad \forall i_1, i_2 \tag{7}$$

$$x^{q}_{i_1,\,i_2} \geq 0, \;\; x^{q}_{i_1,\,i_2}\; integer \quad \forall i_1, i_2 \tag{8}$$

$$0 \leq x_{i_1,\,i_2} \leq 1, \;\; x_{i_1,\,i_2}\; integer \quad \forall i_1, i_2 \tag{9}$$

It can be inferred from the earlier definitions that, the sum of commodities p and q on any arc $(i_1,\ i_2)$ should both equal $(S+R)$ for any feasible solution (recall that $(S+R)$ is the total task size incl. relocations but excl. the virtual task $i=0$). Constraints (2) and (7) are used to ensure the feasibility of flow of commodity p, while Constraints (3) and (8) are for the feasibility of commodity q. Constraint (4) is for tour feasibility. Constraint (5) ensures that the sum of commodities equals $(S+R)$ if the arc $(i_1,\ i_2)$ is in the schedule (thus task i_2 is visited following the completion of task i_1). The precedence between storage tasks is ensured by constraint (6). Constraint (9) is for the binary variables just as in Problem 1a.

Evaluation:

In the above Two Commodity Network Flow formulation for the TSP problem with precedencies, $(S+R+1)^2$ binary variables plus $2(S+R+1)^2$ integer variables are defined, and totally $3(S+R+1)^2 + 3(S+R+1) + (S-1)$ constraint rows are formulated to find the optimal solution (in which only the last $(S-1)$ rows are for the precedence constraints). Given a task set with 10 storages and 10 retrievals, these will be 1323 decision variables and 1395 constraint rows (incl. 9 precedence rows) – with problems of this size, the computational time ranges from 20-ish seconds to over a few minutes to solve the problem on the 2.60GHz, 8GB RAM test computer (over 20 times of that for Problem 1a), which is impractical to be implemented for real-time control of the SBS/RS. Dealing with the precedencies is very costly in terms of computational effort: note that both the size of the DVs and the size of the constraint rows are about tripled in this formulation comparing with those in the MTZ-TSP formulated in Problem 1a,

indicating the fact that handling the task precedencies introduces much computational efforts in finding the optimal schedule of the TSP.

Just like in Problem 1a, the main purpose of the problem formulation here is to provide baselines and insights for the control strategy development. Again, three Dynamic Dispatching policies are tested and compared to optimization results to explore the potential of DD approaches: FCFS-DC policy, Closest First policy, and A* Search Algorithm. The former two policies are the same as those in Problem 1a except for that the storage precedencies are considered in each search. The A* Search Algorithm can be viewed as an extension of the Dijkstra's Algorithm introduced in Problem 1a, because the original Dijkstra's Algorithm is not effective with the presence of precedencies based on our observations and tests. In the Dijkstra's Algorithm implementation in the previous problem, after the completion of task i_1, the next task i_2 is selected based on the tentative distance function $f(i_2) = c_{i_1,i_2} + \min(c_{i_2,i_3})$, where $i_3 \in \{Remaning\ Tasks\}$ and $i_3 \neq i_2$. In the A* Search Algorithm implementation here, the tentative distance function is modified as $f(i_2) = g(i_2) + h(i_2)$, where $g(i_2) = c_{i_1,i_2}$ and $h(i_2)$ is a heuristic function which estimates the remaining makespan starting from the candidate task i_2 by temporarily assuming that the Closest First policy is applied for the rest of the schedule (with storage precedencies considered). The A* Search Algorithm will be improved gradually in the later problem formulations, and the algorithm will be illustrated in detail later in Section 5.5 where the development of the DD-based strategies is further discussed. The optimization and DD results of different task sizes are illustrated in Figure 5.6.

It can be observed that the results of both the A* algorithm and the Closest First policy are very close to the optimization results: the average gaps are 0.8% and 2.4%, respectively. The average gap between FCFS-DC and optimization is 10.3%. The A* Search Algorithm appears to be the best one among all three DD approaches for the current problem formulation. In addition, the A* Search Algorithm is potentially flexible in accommodating more complex decision making in the operational control of practical systems (as opposed to the simplified systems presented in the problem formulations in this section) by integrating various system state indicators into the algorithm formulation. Thus, the A* Search Algorithm viewed as another promising option for the control strategy development, especially with the presence of predetermined storage assignment and storage task precedencies.

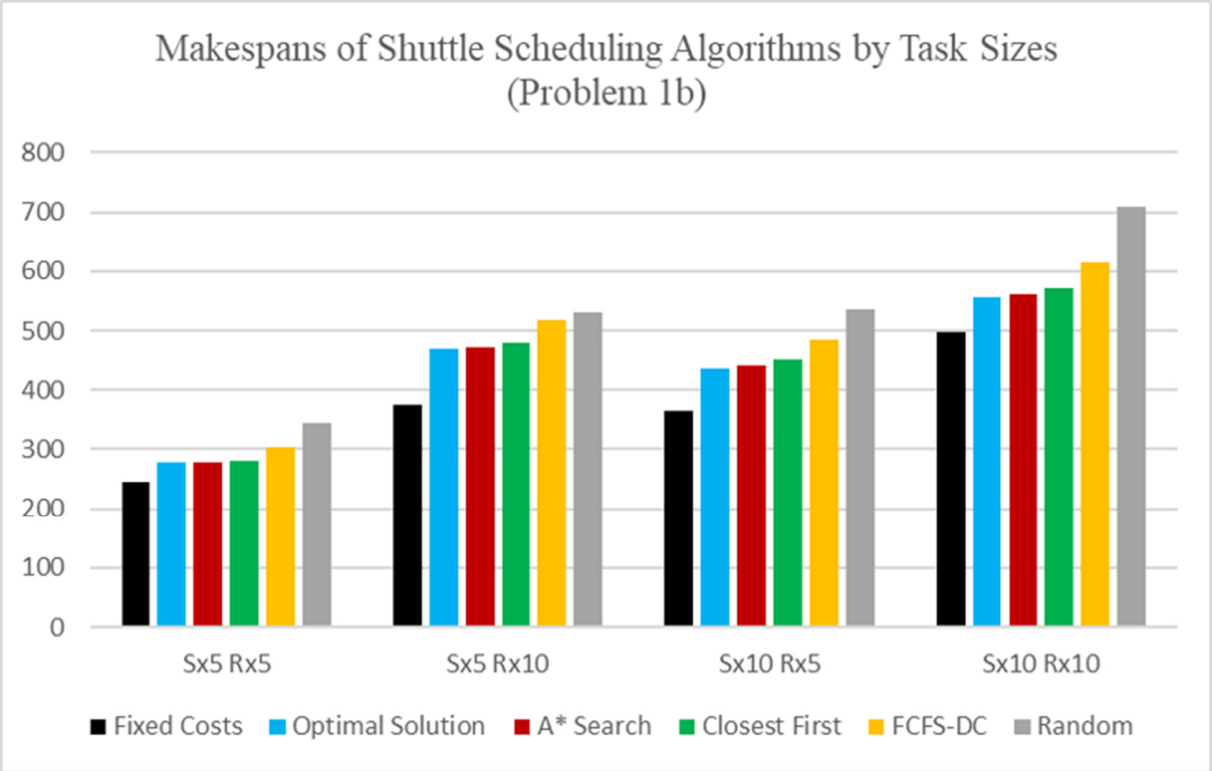

Figure 5.6 Dynamic Dispatching Results vs. Optimal Solution (Problem 1b, algorithm averages of 10 problems for each task size)

Problem 1c: Dedicated blocker-relocation for retrieval tasks

If the tote of a retrieval task is located at a blocked 2-deep slot, the blocker tote must be relocated before accessing the retrieval tote. The shuttle cannot start retrieving a blocked tote prior to the completion of the relocation process. A *relocation task* is defined as the procedure that the shuttle moves to the blocker's column, loads the blocker, moves to the column of the relocation target slot, and unloads the tote. Thus, unlike the formulations in Chapter 4 relocation processes are now viewed as individual tasks instead of components of their corresponding retrieval tasks. Another difference from the approaches in Chapter 4 is that, in the problem formulated here each retrieval task does not need to be made directly following its corresponding relocation task: instead, it can be started at a later point in the schedule as long as the relocation task is completed – such relaxation is made to discover any potential opportunities in improving the scheduling performance. With these assumptions, the relocation-retrieval relationships are described as task precedencies just like our approaches in Problem 1b.

For the simplicity of explanation, it is assumed here that the storage totes are indistinctive just like in Problem 1a, thus the relocation-retrieval relationships are the only precedencies considered in this problem. Also, it is assumed that each blocked tote is blocked by an existing blocker tote instead of by an incoming storage tote, and each blocker tote is not requested by any other retrieval task – if these two assumptions do not stand, then additional task precedencies need to be considered. This Problem 1c is formulated by modifying the Problem 1a formulations as follows:

Problem Inputs:

S, R, $\boldsymbol{A_S}$, y_0: Same definitions as in Problem 1a.

Denote Re as the number of relocation tasks, where $Re \leq R$.

i : Task index, where $i \in \{0, 1 \cdots (S + R + Re)\}$. The index $i = 0$ indicates the start point of the schedule. The subset $\{1, 2 \cdots S\}$ are for the S storage tasks. It is assumed that the retrieval task set is sorted so that the first Re tasks in the set are those which require relocations, hence the subset $\{(S + 1), (S + 2) \cdots (S + Re)\}$ are for the Re retrieval tasks which require relocation, and the subset $\{(S + Re + 1), (S + Re + 2) \cdots (S + R)\}$ are for the $(R - Re)$ retrieval tasks not require relocation. Finally, the last subset $i \in \{(S + R + 1), (S + R + 2) \cdots (S + R + Re)\}$ are the indices of the relocation tasks, where each i corresponds to the retrieval task of index $(i - R)$.

π_i and $[x_i, y_i, z_i]$: For each relocation task i, define its task type $\pi_i = 3$ and its target slot $[x_i, y_i, z_i]$ indicating the slot to which the blocker tote will be relocated. For storage and retrieval tasks the definitions are the same as in Problem 1a.

$\boldsymbol{A_{Re}}$: A set of slot locations selected for the Re blocked retrieval tasks according to some given storage assignment policy. The size of the set $|\boldsymbol{A_{Re}}| = Re$. For the simplicity of problem formulation, unlike the assumption made in Problem 1a where all storage totes are indistinctive, here each relocation target slot $[x_i, y_i, z_i] \in \boldsymbol{A_{Re}}$ is assumed pre-determined for each relocation task $i \in \{(S + R + 1), (S + R + 2) \cdots (S + R + Re)\}$, and $\boldsymbol{A_{Re}} \cap \boldsymbol{A_S} = \emptyset$. Like in the previous problems, this assumption is also based on the fact that the number of available slots is usually much larger than the number of waiting tasks on a tier. Each relocation task is either a direct predecessor or indirect predecessor of the corresponding retrieval task. A schedule is feasible if

each retrieval is performed later than the corresponding relocation. Thus, the $S+R$ tasks construct a $(S+R+Re+1)$ nodes' Travelling Salesman Problem with precedence constraints. The travel time elements are then modified based on their formulations in Problem 1a (note that only the modified formulations are showed).

d_i : The remaining service time of task i from the time the shuttle arrived its load point, computed as:

$d_0 = 0$, because the initial state of the shuttle is assumed idle;

$$d_i = \begin{cases} \tau^V_{0,\,y_i} + \omega^V_0 + \omega^V_{z_i}, & if\ \pi_i = 1\ or\ 2 \\ \tau^V_{y_{(i-R)},\,y_i} + \omega^V_0 + \omega^V_{z_i}, & if\ \pi_i = 3 \end{cases}$$

The fixed costs $D = \sum_{i=1}^{S+R+Re} d_i$

$c_{i_1,\,i_2}$: Shuttle travel time from the start location (load point) of task i_1 to the start location (load point) of task i_2. Obviously, the load point of each relocation task i is always the same as the target column of its corresponding retrieval task $(i-R)$, thus the travel times involving relocations can be formulated as followed:

$If\ i_1 = 0\ (thus\ i_2\ is\ the\ first\ task\ in\ schedule)$:

$$c_{i_1,\,i_2} = \tau_{0,\ i_2} = \tau^V_{y_0,\,y_{(i_2-R)}} \quad if\ \pi_{i_2} = 3 \quad (Reallocation)$$

$Otherwise$:

$$c_{i_1,\,i_2} = \begin{cases} \tau^V_{y_{i_1},\,y_{(i_2-R)}} & if\ \pi_{i_1} = 1, \pi_{i_2} = 3 & (S-Re\ case) \\ \tau^V_{0,\,y_{(i_2-R)}} & if\ \pi_{i_1} = 2, \pi_{i_2} = 3, i_1 \neq i_2 + R & (R-Re\ case) \\ M & if\ \pi_{i_1} = 2, \pi_{i_2} = 3, i_1 = i_2 + R & (Infeasible) \\ \tau^V_{y_{(i_1-R)},\,0} & if\ \pi_{i_1} = 3, \pi_{i_2} = 1 & (Re-S\ case) \\ \tau^V_{y_{(i_1-R)},\,y_{i_2}} & if\ \pi_{i_1} = 2, \pi_{i_2} = 2 & (Re-R\ case) \\ \tau^V_{y_{(i_1-R)},\,y_{(i_2-R)}} & if\ \pi_{i_1} = 3, \pi_{i_2} = 3 & (Re-Re\ case) \end{cases}$$

Where M is a sufficiently large number to indicate infeasible retrieval-relocation paths due to blocking. The travel time elements for other service cases remain the same as in Problem 1a.

Decision Variables:

Just like in Problem 1b, Two Commodity Network Flow formulation is applied to handle the precedence constraints between retrievals and relocations. The binary variables x_{i_1, i_2} and integer variables $x^p_{i_1, i_2}$ and $x^q_{i_1, i_2}$ are defined in the same way as in Problem 1b, except for that all of them are $(S + R + Re + 1) \times (S + R + Re + 1)$ matrices instead of $(S + R + 1) \times (S + R + 1)$ matrices here.

Objective Function:

The objective function of Problem 1c is the same as that of Problem 1b except for that all $(S + R)$ are replaced by $(S + R + Re)$, as follows:

$MIN\ \ F(x)$, where:

$$F(x) = \sum_{i_1=0}^{S+R+Re} \sum_{\substack{i_2=0, \\ i_2 \neq i_1}}^{S+R+Re} \left(c_{i_1, i_2} \times \frac{x^p_{i_1, i_2} + x^q_{i_1, i_2}}{S + R + Re} \right) \quad (1)$$

Like in Problem 1a and 1b, the later part of the objective function is a constant because all d_i's are computed previously.

Constraints:

Constraints (2) to (5) are the same as those in Problem 1b except for that all $(S + R)$ are replaced by $(S + R + Re)$. Constraint (6) which was for the storage precedencies in Problem 1b is modified as followed for the relocation-retrieval precedencies:

$$\sum_{i_2=0}^{S+R+Re} x^p_{i_1+R, i_2} - \sum_{i_2=0}^{S+R+Re} x^p_{i_1, i_2} \geq 1 \quad \forall i_1 \in \{(S + 1), (S + 2) \cdots (S + Re)\} \quad (6)$$

Constraints (7) to (9) are the same as those in Problem 1b.

Evaluation:

In the above Two Commodity Network Flow formulation for the TSP problem with precedencies, $(S + R + Re + 1)^2$ binary variables plus $2(S + R + Re + 1)^2$ integer variables are defined, and totally $3(S + R + Re + 1)^2 + 3(S + R + Re + 1) + Re$ constraint rows are formulated to find the optimal solution (in which only the last Re rows are for the precedence

constraints) – these numbers are typically more than those in Problem 1b because the nodes size here is increased from $(S + R + 1)$ to $(S + R + Re + 1)$ even when the relocations precedencies (Re) are relatively few and the storage precedencies are not concerned. Given a task set with 10 storages and 10 retrievals among which 4 retrievals require relocations, that will be 1875 decision variables and 1954 constraint rows (incl. 4 precedence rows) – with problems of this size, the computational time ranges from 30-ish seconds to a few minutes to solve the problem on the same 8GB RAM computer in the previous problems, which is again impractical to be implemented for real-time control of the SBS/RS. Thus, just like in Problem 1b, the optimization approach formulated here is viewed as impractical for real-time control of the SBS/RS.

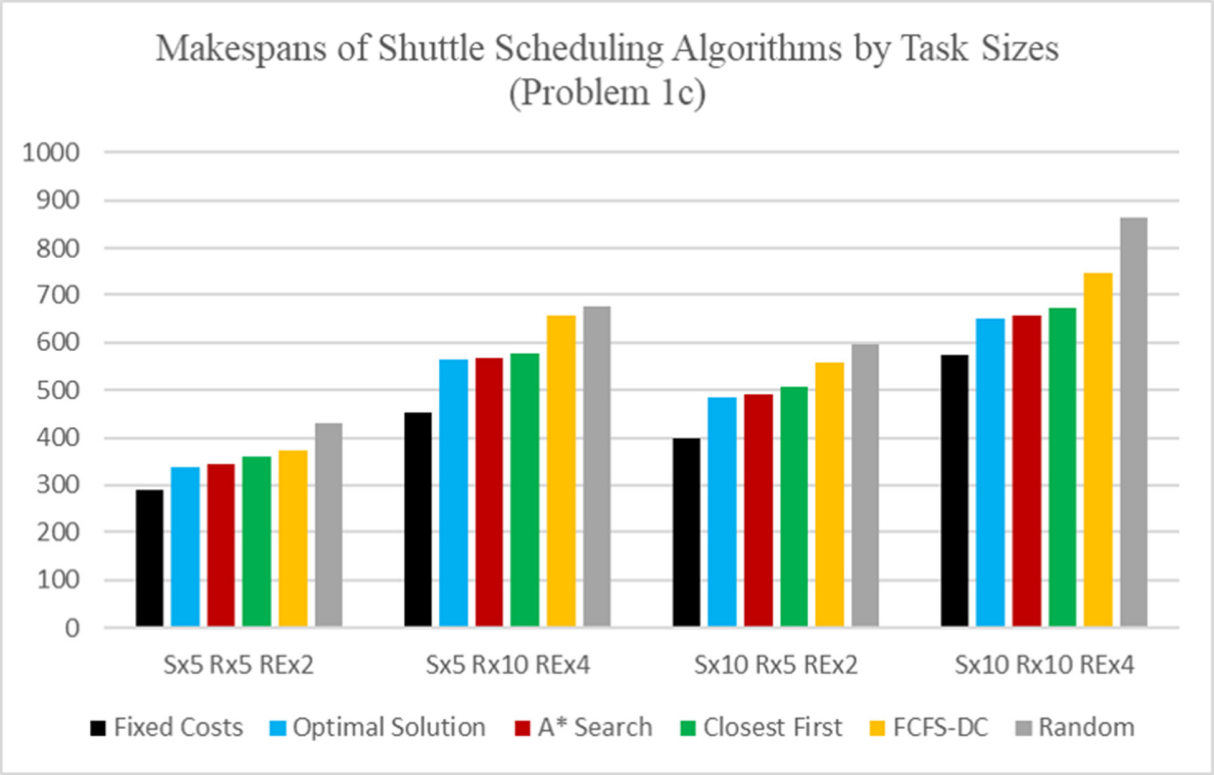

Figure 5.7 Dynamic Dispatching Results vs. Optimal Solution (Problem 1c, algorithm averages of 10 problems for each task size)

Like in the previous problems, three Dynamic Dispatching policies are tested and compared to optimization results to explore the potential of DD approaches: FCFS-DC policy, Closest First policy, and A* Search Algorithm. All the DD approaches obtain their solutions in the same ways as in Problem 1b, except for that the relocation tasks are treated as additional tasks and

precedence constraints are considered for the relocation-retrievals instead of for the storages (for FCFS-DC, each relocation task is performed right before its corresponding retrieval task). The optimization and DD results of different task sizes are illustrated in Figure 5.7. The number of relocation tasks are determined as 40% of the number of retrieval tasks (which is the expected value of relocation ratio θ^R given the average inventory level is 83.3% according to the analytical approaches and assumptions in Section 4.4.

It can be observed that the results of both the A* algorithm and the Closest First policy are very close to the optimization results: the average gaps are 0.9% and 3.9%, respectively – both outperformed the FCFS-DC policy (average gap is 14.1%) especially for larger problem sizes. Like in the previous problems, the A* Search Algorithm is again considered as an attractive option for the control strategy development.

Problem 1d: Precedencies for both dedicated storage and dedicated blocker-relocation

With the presence of both types of precedencies described in Problem 1b and 1c, the problem can still be formulated with the same Two-Commodity Network Flow approach. The problem inputs, decision variables and objective function are the same as in Problem 1c. The constraints are also the same except for the precedence constraint (6), which is formulated for both precedence constraint types as follows:

$$\begin{cases} \sum_{i_2=0}^{S+R} x^p_{i_1-1,\,i_2} - \sum_{i_2=0}^{S+R} x^p_{i_1,\,i_2} \geq 1 \quad \forall i_1 \in \{2,3\cdots S\} \\ \sum_{i_2=0}^{S+R+Re} x^p_{i_1+R,\,i_2} - \sum_{i_2=0}^{S+R+Re} x^p_{i_1,\,i_2} \geq 1 \quad \forall i_1 \in \{(S+1),(S+2)\cdots(S+Re)\} \end{cases} \tag{6}$$

Evaluation:

In this problem, $(S+R+Re+1)^2$ binary variables plus $2(S+R+Re+1)^2$ integer variables are defined, and totally $3(S+R+Re+1)^2+3(S+R+Re+1)+(S-1+Re)$ constraint rows are formulated to find the optimal solution (in which only the last $(S-1+Re)$ rows are for the precedence constraints). Comparing with Problem 1c, the DVs are the same and the constraint rows are just slightly more due to the additional $(S-1)$ constraint rows for storage

precedencies. Given a task set with 10 storages and 10 retrievals among which 4 retrievals require relocations, there will be 1875 decision variables and 1963 constraint rows (incl. 13 precedence rows), thus the optimization approach is again viewed as impractical for real-time system control because the computation times are in general over a few minutes (8GB RAM computer) for this problem size. Three Dynamic Dispatching policies that consider both storage precedencies and relocation precedencies are tested and compared with optimization results, as illustrated in Figure 5.8. The performances of the DD policies are similar to those in Problem 1b and 1c in which only one type of precedencies is considered. The average gaps of A* algorithm and the Closest First policy results to optimization results are 0.8% and 3.4%, respectively, both outperformed the FCFS-DC policy (average gap is 12.9%) especially for larger problem sizes.

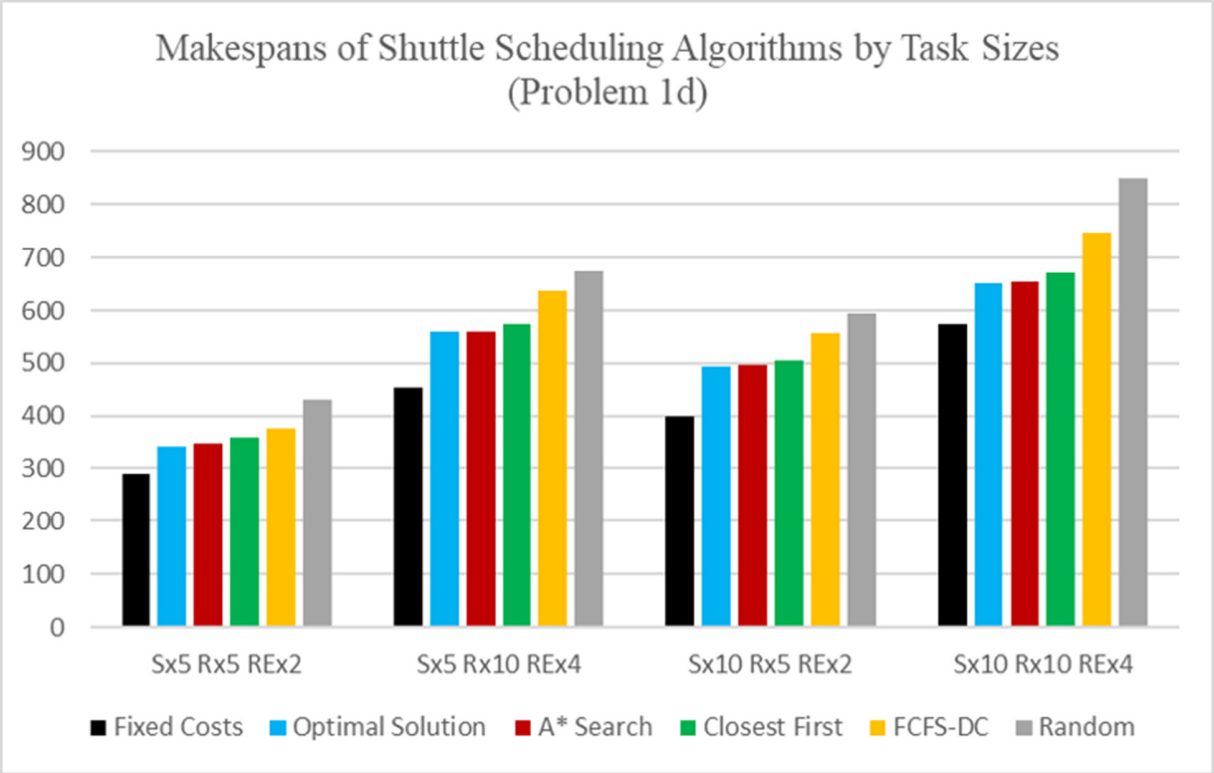

Figure 5.8 Dynamic Dispatching Results vs. Optimal Solution (Problem 1d, algorithm averages of 10 problems for each task size)

5.4.2 Scheduling of multiple shuttles and different types of lifts

The device scheduling problem gets much more complicated when the scope is expanded from a single shuttle to all devices in the aisle, mainly caused by the additional task precedencies

between different service stages. More specifically, a storage task cannot be served by the shuttle before its service by the storage lift is completed, and similarly a retrieval task cannot be served by a retrieval lift before its service by the shuttle is completed. Moreover, the touring service patterns of the multi-unit tote lifts and the roller-conveyor characteristics of both input buffers and output buffers introduce additional complexity to the scheduling problem.

Problem 2a: Single shuttle scheduling with storage time windows

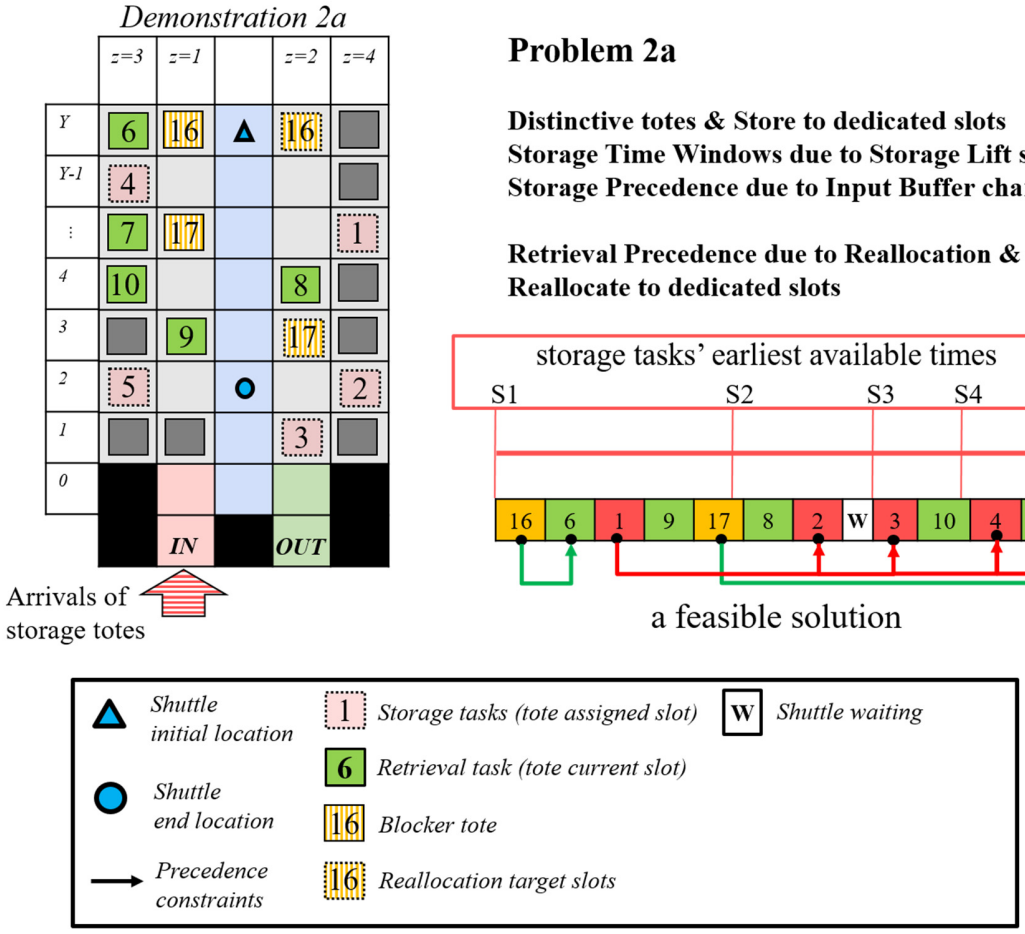

Figure 5.9 Problem formulations of shuttle scheduling in Tier-captive SBS/RS (2a)

The storage totes are delivered to the target tiers by the storage lift. The shuttle cannot start serving a storage task until the storage tote arrives at the exit of the input buffer of the tier. Such lift-shuttle interaction determines the earliest available time for each storage task for shuttle service. As illustrated in the example in Figure 5.9, the start time for each storage task cannot be earlier than the earliest available time for shuttle service, and waiting times may occur in the

shuttle's schedule (between task 2 and 3 in Figure 5.9).We observed such complexity resembles the Traveling Salesman Problem with Time Windows (TSPTW). In TSPTW, a time window $[a_i, b_i]$ is described for each node i to be visited as problem input. A schedule is feasible only if each node i is visited no earlier than its earliest visit time a_i and no later than its latest visit time b_i. The various precedence constraints discussed in Problem 1b, 1c and 1d can all be integrated to TSPTW formulations, thus there is no need to apply the Two-Commodity Network Flow formulation if time windows are involved. On the other hand, the TSPTW is much more computationally costly than the Two-Commodity Network Flow TSP. Thus, in this section, we explore the TSPTW formulation only to provide baselines for the design and evaluation of Dynamic Dispatching approaches.

Decision Variables:

The Asymmetric TSPTW formulation proposed by Kara and Derya (2015) is applied. Like in the previous problems, define x_{i_1, i_2} as a $(S + R + Re + 1) \times (S + R + Re + 1)$ matrix of binary variables (BVs) indicating if task i_2 is the next task after task i_1 in the shuttle's schedule, where $i_1, i_2 \in [0, 1, 2 \dots S + R]$. Then, denote t_i as an $(S + R + Re)$ array of numeric variables indicating the start time of task $i \in \{1, 2 \dots S + R + Re\}$. At last, define t_{end} as a numeric variable indicating the completion time of the schedule.

Objective Function:

Considering the possible waiting times in the schedule, the objective function (total makespan) cannot be formulated in the $\sum\sum(c_{i_1, i_2} x_{i_1, i_2})$ forms as in Problem 1. Instead, it is formulated as the completion time of the schedule, as follows:

$MIN\ F(x)$, where:

$$F(x) = t_{end} \tag{1}$$

Constraints:

$$t_i - c_{0,i} x_{0,i} \geq 0, \qquad \forall i > 0 \tag{2}$$

$$t_i \geq a_i, \qquad \forall i > 0 \tag{3}$$

$$t_i \leq b_i, \qquad \forall i > 0 \tag{4}$$

$$t_{i_1} - t_{i_2} + \left(b_{i_1} - a_{i_2} + c_{i_1, i_2}\right) x_{i_1, i_2} \leq b_{i_1} - a_{i_2}, \qquad \forall i_1, i_2 > 0, i_1 \neq i_2 \tag{5}$$

$$\begin{cases} t_i \geq t_{i-1} + d_{i-1}, & \forall i \in \{2, 3 \dots S\} \\ t_i \geq t_{i+R} + d_{i+R}, & \forall i \in \{S+1, S+2 \dots S+R+Re\} \end{cases} \tag{6}$$

$$t_i + c_{i,0} \leq t_{end}, \qquad \forall i > 0 \tag{7}$$

$$\sum_{i_2=0}^{S+R+Re} x_{i_1, i_2} \leq 1, \qquad \forall i_1 > 0 \tag{8}$$

$$\sum_{i_1=0}^{S+R+Re} x_{i_1, i_2} \leq 1, \qquad \forall i_2 > 0 \tag{9}$$

$$0 \leq x_{i_1, i_2} \leq 1, x_{i_1, i_2}\ integer \quad \forall i_1, i_2 \tag{10}$$

Constraint (2) guarantees that the start time of each task cannot be earlier than time zero plus the shuttle travel time from the shuttle's initial location. Constraint (3) and (4) guarantee that the start times of tasks are within the corresponding time windows. Constraint (5) ensures $t_{i_2} \geq t_{i_1} + c_{i_1, i_2}$ if the arc $(i_1,\ i_2)$ exist in the schedule. Also, constraint (5) guarantees that start times in the schedule constitute an increasing step function, which means that these inequalities prohibit illegal subtours, i.e., they are the subtour elimination constraints of our formulation. Constraints (6) are precedence constraints, where d_{i-1} and d_{i+R} are the service times of the corresponding predecessors for storages and relocations, respectively. Constraint (7) guarantees that the makespan is less than or equal to the start time plus the service time of last task in the schedule. Constraints (8), (9) and (10) are standard constraints for the binary variables.

Evaluation:

In this problem, $(S + R + Re + 1)^2$ binary variables plus $(S + R + Re + 1)$ numeric variables are defined. Totally $2(S + R + Re + 1)^2 + 5(S + R + Re + 1) + (S - 1 + Re)$ constraint rows are formulated to find the optimal solution, which looks at the first glance simpler than the Two-Commodity Network Flow formulation in Problem 1d. However, based on our tests on the same 2.60GHz clock speed, 8GB RAM computer for testing the previous problem formulations, we found that the TSPTW formulated here is very computational expensive – even with the smallest task size (5 storages, 5 retrievals, 2 relocations), in most cases it took more than

10 minutes for the TSPTW to find the optimal solution. We speculate that it is because the TSPTW becomes a Mixed Integer Programming problem where both integer variables and non-integer variables exist, which increases the solver's iteration efforts significantly (note that all the previous problems only have integer variables). Because the computation time for larger problem sizes could be extremely large, we determine a time limit of 600,000 milliseconds (10 minutes) for the optimization solver – we assume it is adequate for the solver to obtain at least "satisfactory" solutions, based on the assumption that the solution space is relatively smooth considering the rectilinear pattern of shuttle services. The optimal or near-optimal makespan from the TSPTW is collected for each task set, and viewed as the baseline for evaluating Dynamic Dispatching performances.

Just like in the previous problems, Random Scheduling, FCFS-DC, Closest First, and A* Search Algorithm are determined as the DD approaches to be evaluated, where the next task to the schedule is selected once the previous task is completed. The Random Scheduling, FCFS-DC and Closest First policies select the next task in the same way as in Problem 1d, while the search space is limited to the tasks currently available by the time the previous task is completed. If no tasks are available at that time, the shuttle waits until at least one task becomes available. The A* Search Algorithm implemented here is more complex than its implementations in the previous problems. Once a task i_1 is completed, the tentative distance function of a candidate task i_2 is presented as $f(i_2) = g(i_2) + h(i_2)$, where $g(i_2)$ is the time costs c_{i_1, i_2} from the start location (unload point) of i_1 to the start location (load point) of task i_2 (incl. the service time of i_1) plus the waiting time incurred if i_2 is not available yet, and $h(i_2)$ is a heuristic function that estimates the remaining makespan starting from the candidate task i_2 by temporarily assuming that the Closest First policy is applied for the rest of the schedule (with available times and precedencies considered). The candidate with the smallest tentative distance is then selected as the next task. The details of the A* Search Algorithm will be introduced later in Section 5.5 where the development of the DD-based strategies is further discussed.

Four task sizes are decided to test the optimization and DD approaches, and 10 task sets are generated randomly for each task size. Note that the optimization results found by the TSPTW are not necessarily true optimal results, because the solver time is limited 10 minutes. It is observed that with larger task sizes, the optimization results found within the time limits could be worse

than the best solutions found through the DD approaches. Particularly for the largest task size (10 storages, 10 retrievals, 4 relocations, totally 25 nodes incl. the virtual task 0), the optimization results found are significantly worse than the best DD results. We extended the solver time limits for this problem size to 60 minutes and updated the results, but that seems not very helpful in finding the optimal solutions.

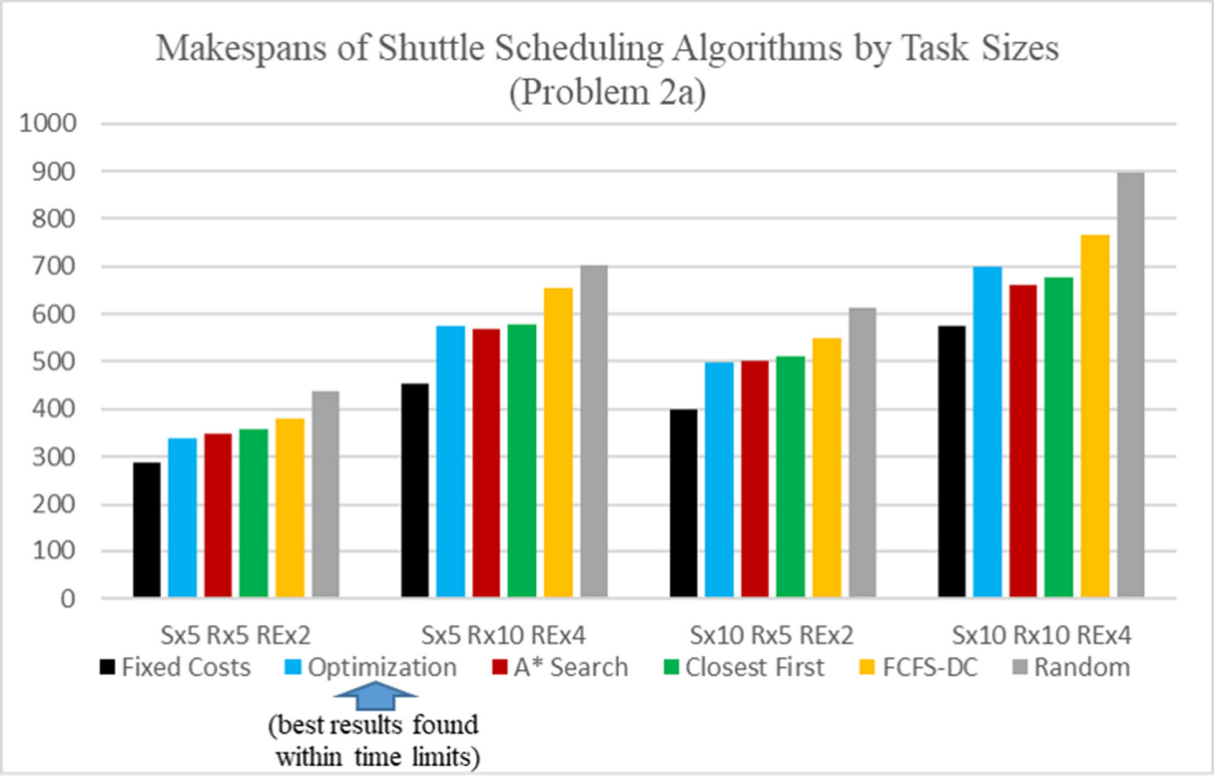

Figure 5.10 Dynamic Dispatching Results vs. Optimization Results (Problem 2a, algorithm averages of 10 problems for each task size)

The optimization and DD results (averages for 10 task sets from each task size) are illustrated in Figure 5.10. Although the optimization results are no longer exact baselines as in previous problems, the Closest First policy and A* Search Algorithm results are viewed as satisfactory by compared to the optimization results found and, the FCFS-DC and Random Scheduling results. Moreover, the A* Search Algorithm (adapted for the time window constraints here) appears to perform better than the Closest First policy, which is consistent with our observations in the previous problem formulations. We later rerun the experiments using Ant Colony Optimization (ACO, which is a heuristic optimization approach, to be introduced in detail

later this section) and found that the average gaps in shuttle makespan are within 1% between the A* Search Algorithm results and the ACO results. Hence, the A* Search Algorithm approach for single shuttle scheduling is viewed as not only effective in obtaining near-optimal results but also robust for various system/task assumptions. We also speculate that A* Search Algorithm approach is potentially adaptive for various operational environments. Thus, the A* Search Algorithm is determined as the prototype DD policy for shuttle scheduling, assuming that the development of the shuttle scheduling policy will be based on the fine-tuning of the A*'s tentative distance function $F(i) = G(i) + H(i)$. In the later sections, this development methodology will be further explored and evaluated together with the lifts scheduling approaches as well as storage assignment approaches under various demand scenarios.

Problem 2b: Scheduling all devices in aisle for Tier-captive SBS/RS

The scheduling problem for all devices of the SBS/RS aisle is much more complicated because the schedule for each device needs to be specified (as discussed in Section 5.2 and illustrated in Figure 5.2). Let us temporarily assume the total makespan of all tasks in a given task set is the objective function to be minimized and assume that the tasks are deterministic and that no new tasks will be added to the task set before the completion of the task set. As all devices involved in serving the task set are considered, the total makespan is now defined as the completion time of the last task in the task set. Obviously, the total makespan here considers not only each task's service time in each service stage by the corresponding device, but also its possible waiting times between service stages (on the input or output buffer). Moreover, because the devices (shuttles and tote lifts) serve their own parts of tasks in parallel, the completion time of the task set can be either one of the two cases: 1) the time when the retrieval lift completes the last retrieval task from the task set and then the set becomes empty, or 2) the time when one of the shuttles completes the last storage task from the task set and then the set becomes empty. Due to this fact, the total makespan cannot be formulated in the relatively simple TSP forms as in the previous problems for single shuttle scheduling. Also, precedence and time window constraints exist between shuttle services and tote lift services. The objective function for minimizing the overall makespan can be presented at high level as follows:

$$MIN \max\left[F^{RL}, max\left(F_j^V\right)\right] \quad (1)$$

In the above objective function, F^{RL} is the makespan of the retrieval lift's schedule, and F_j^V is the makespan of the schedule of each shuttle $j \in \{1,2 \dots J\}$.

The schedule of the retrieval lift is described in two aspects: the sequences and timings that the lift serves the retrieval tasks completed by the shuttle, and the tours of those tasks. It is noticeable that the retrieval lift's task sequences need to be consistent with the schedules of all the shuttles involved for serving the task set. Figure 5.11 illustrates such consistency constraints, for example, the retrieval lift cannot serve task 13 before task 12, because task 13 is served by the shuttle (V4) after task 12. As the I/O buffers are assumed to be roller-conveyors, task 13's tote cannot arrive the exit of the output buffer until task 12's tote is loaded by the retrieval lift.

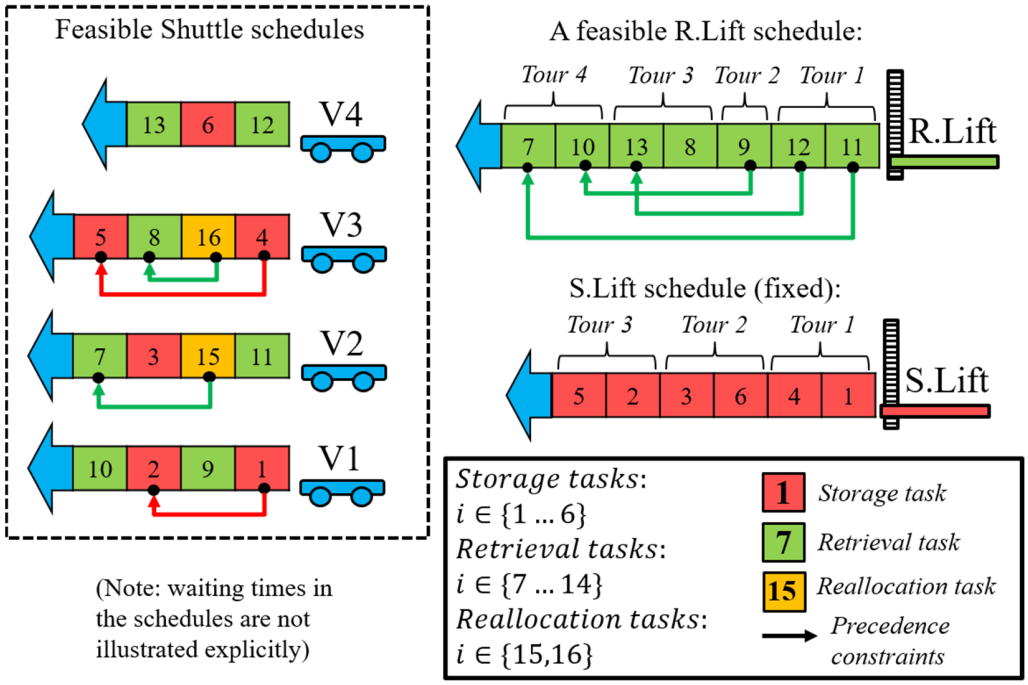

Figure 5.11 Problem formulation of all-device scheduling in Tier-captive SBS/RS (2b)

An Ant Colony Optimization (ACO) heuristic optimization approach is applied for this problem. ACO algorithm is a probabilistic technique for solving computational problems that can be reduced to finding good paths through graphs, where multi-agent methods inspired by the pheromone-based communication behavior of real ants are used to solve numerous optimization problems (Monmarché, et al, 2010). In our implementation of ACO approach, the task sequence of the retrieval lift is determined as the solution of the problem – note that the task sequence is not

equivalent to the retrieval lift's schedule, because the tasks' completion times from the upstream shuttle services are not yet known. Instead, the schedules and makespan of each device are derived from a retrieval task sequence which is defined as a "*Solution*" under specific system and control assumptions – to be introduced later.

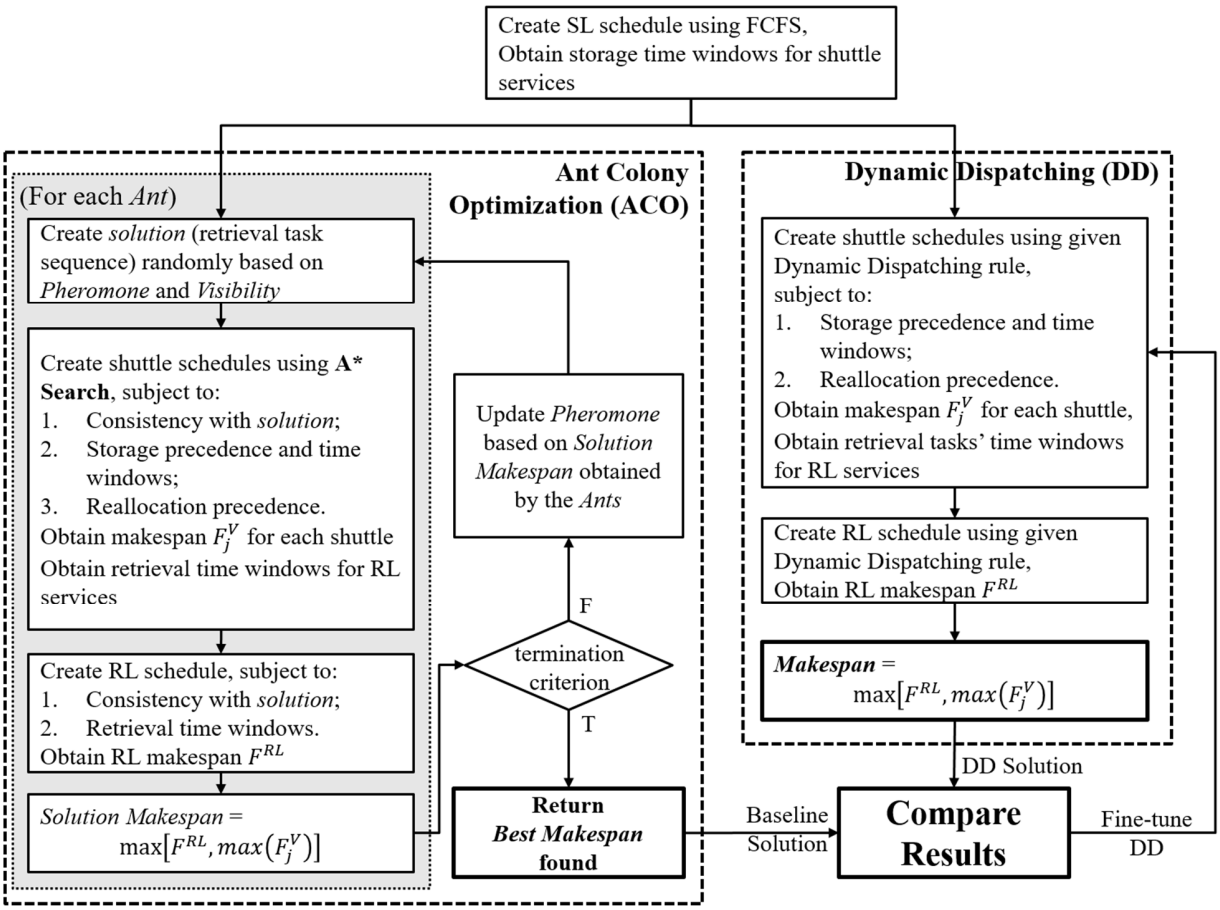

Figure 5.12 Ant Colony Optimization Implementation and Development of Dynamic Dispatching Rules

Although ACO is a heuristic optimization approach that cannot guarantee optimality, the ACO results obtained for reasonable problem sizes and under proper algorithm settings are assumed to be near-optimal. Thus, the ACO results are used as baselines for the development of Dynamic Dispatching rules. Our ACO implementation is illustrated in Figure 5.12. Simplification assumptions are made for the devices in the aisle. It is assumed that the retrieval lift selects the next retrieval task according to the task sequence solution found by the current ant, and if the next task in sequence is not yet completed shuttle service, the retrieval lift stays idle until its completion.

It is assumed that storage totes going to the aisle are delivered to the load point of the storage lift by a single roller conveyor, and the target slots of each storage tote are predetermined before storage lift services. In addition, it is assumed that the 2-capacity tote lifts (either type) always perform full-size tours (this assumption is reasonable here because the objective is minimizing the overall makespan). Finally, the capacities of the I/O buffer on tiers are assumed to be adequately large, thus the service flows from upstream to downstream will not be suspended due to the buffer sizes.

Under the above assumptions, the schedule of the storage lift is deterministic given a specific task set, and thus the storage time windows for shuttle tasks are known at the beginning. It is assumed that the shuttle selects its next task based on the A* Search Algorithm (which is viewed as near-optimal for single shuttle scheduling in the previous problems), while the search space is subject to three types of precedence constraints:

1) Storage precedence and time windows as introduced in Problem 1b and 2a, respectively;
2) Relocation-retrieval precedence as introduced in Problem 1c;
3) Consistency with the retrieval task sequence in the solution.

The makespan of each shuttle schedule (F_j^V) is thus obtained. Finally, the completion time of retrieval tasks from shuttle services are recorded, and their available times (release times) for retrieval lift service are obtained by adding the fixed transportation time on the output buffer conveyors.

The retrieval lift selects its next task in one of the three cases: 1) a retrieval task is just released for retrieval lift service when the retrieval lift is idle; 2) the retrieval lift has just completed its current tour and there exists at least one waiting task to be added to a new tour; 3) the retrieval lift has just loaded the first task in tour and there exists at least one waiting task to be added to the same tour. In our ACO implementation, the retrieval lift schedule is constructed where the next task is the first available task from the remaining tasks according to the solution sequence. Thus, given the solution and the retrieval tasks' available times obtained in the previous step, the retrieval lift's schedule can also be developed deterministically, and the retrieval lift's makespan (F^{RL}) is obtained. Finally, each ACO solution is mapped to deterministic device schedules as well as the overall makespan, as illustrated in Figure 5.13.

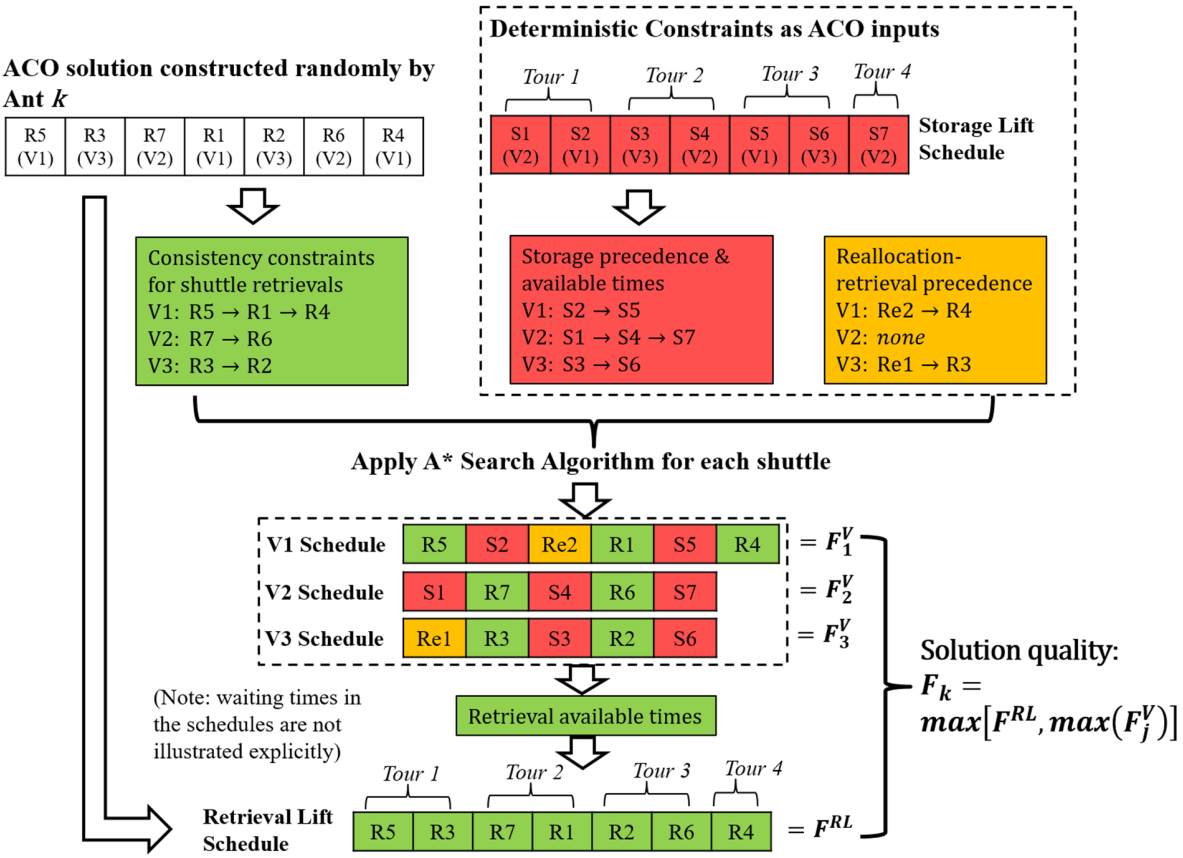

Figure 5.13 Mapping an ACO solution to device schedules

Problem Inputs:

S : Number of total storage tasks;

R : Number of total retrieval tasks;

Re : Number of total relocation tasks;

J : Number of shuttles;

S_j : Number of storage tasks to shuttle j, there is $\sum_{j=1}^{J} S_j = S$;

R_j : Number of retrieval tasks to shuttle j, there is $\sum_{j=1}^{J} R_j = R$;

Re_j : Number of relocation tasks to shuttle j, there is $\sum_{j=1}^{J} Re_j = Re$ and $Re_j < R_j, \forall j$;

In this scheduling problem where multiple devices are considered, some device might be still processing tasks assigned earlier that are not described in the current task set. Thus, the earliest available time for each device must be defined as problem inputs:

$t0_j^V$: The earliest available time of shuttle j;

$y0_j^V$: The initial Y location of shuttle j on its tier (if it is busy, defined as the end location of its current task);

$t0^{SL}$: The earliest available time of the storage lift (if it is busy, defined as the time when it returned to the I/O floor after completed its current tour);

$t0^{RL}$: The earliest available time of the retrieval lift (if it is busy, defined as the time when it completed its current tour at the I/O floor);

t_B : The tote transportation time on each I/O buffer conveyor.

Denote $i \in \{1,2 \dots S\}$ as the storage tasks with fixed sequence for storage lift services, their earliest available times for shuttle services are obtained as follows:

$$a_{i^*}^V = \begin{cases} \tau_{0,x_{i^*}}^{TL} + 3\omega^{TL} + \sum_{i=1}^{i<i^*-1} \left(\tau_{0,x_i}^{TL} + \tau_{x_i,x_{i+1}}^{TL} + \tau_{x_{i+1},0}^{TL} + 4\omega^{TL}\right), if\ i\ is\ odd \\ \tau_{0,x_{i^*-1}}^{TL} + \tau_{x_{i^*-1},x_{i^*}}^{TL} + 4\omega^{TL} + \sum_{i=1}^{i<i^*-2} \left(\tau_{0,x_i}^{TL} + \tau_{x_i,x_{i+1}}^{TL} + \tau_{x_{i+1},0}^{TL} + 4\omega^{TL}\right), if\ i\ is\ even \end{cases}$$

$$+t0^{SL} + t_B$$

In the equations above, it is assumed in the current problem formulations that the next incoming tote to the storage lift is always ready at the input of the system, which means the upstream conveyor system will not cause any delay to the SBS/RS. Relaxation of this assumption can be formulated easily as task time windows for storage lift services, but we do not expand our scope to there because the current problem aims at minimizing the overall makespan assuming all tasks are ready for its first-stage service (lift services as for storages, shuttle services as for retrievals) in the SBS/RS.

ACO Formulation:

In each search iteration, each ant constructs a feasible solution randomly based on both the Visibility matrix and the Pheromone matrix. In our approach, the Visibility matrix V is evaluated dynamically according to the current tour size. The solution is then mapped to the device schedules to obtain the objective function (overall makespan) in three steps. The solutions found in each iteration are then compared to the historical best solutions, and the Pheromone matrix is updated according to the solution qualities. The formulation details of the Ant Colony Optimization heuristics are further described in Appendix II. In addition, multiple ACO replications are performed to solve each problem – the settings are viewed as robust when no significant differences can be observed between the results found by different replications (within $\pm 0.2\%$ of the replication average). As the ACO approach here is designed to provide baselines for the development and evaluation of DD policies (instead of providing direct system control solutions), we will not further explore the approaches for improving the computational efficiency of the proposed ACO algorithm.

Evaluation:

A 16-tier, 200-column, 2-deep Tier-captive configuration is chosen as the system for testing the ACO and the candidate Dynamic Dispatching approaches. Four problem sizes are determined: [75 S, 75 R, 20 Re], [75 S, 100 R, 40 Re], [100 S, 75 R, 30 Re] and [100 S, 100 R, 40 Re], and 10 task sets are generated randomly for each problem size. Note that for each task set, because the target slots are assigned randomly among all slot locations in the rack, the task sizes vary by tier and thus by shuttle.

Two Dynamic Dispatching approaches are determined as the DD candidates under evaluation. In both DD approaches, the shuttle schedules are developed using A* Search Algorithm described in Problem 2a (with storage precedence and time windows and relocation precedence) – note that there are no additional retrieval precedence constraints here, because unlike in the ACO, in the DD approaches, the shuttles' schedules are determined initially, and the retrieval lift services are then scheduled based on the shuttles' schedules. Hence, the two DD candidates only differ from the perspective of retrieval lift scheduling:

1) A*-FCFS: Using A* Search Algorithm for shuttle scheduling, and First-come-first-serve rule for retrieval lift scheduling. The retrieval lift serves tasks by the times the totes arrive to the

exit of the output buffers, and always perform full-size tours until there are no remaining retrieval tasks in the task set;

2) A*-Closest: Using A* Search Algorithm for shuttle scheduling and Closest-First rule for retrieval lift scheduling. The retrieval lift's scheduling decision is triggered in three situations: 1) A tote just arrived the buffer exit when the lift is idle; 2) The lift just completed a tour and there is at least one tote standing by for service; and 3) The lift has just loaded the first tote of its current tour while there is still tour space for a second tote, and there is at least one tote standing by – if there is no tote standing by at the moment, the lift will proceed with single-tote tour. The retrieval lift selects the next task from currently available candidates (totes standing-by at the exit of the output buffer) by the vertical distances from the lift's current location, and break-tie by the original sequence in the task set.

The ACO results are viewed as baselines for the DD approaches. Recall that the objective function is $F = \max\left[F^{RL}, max\left(F_j^V\right)\right]$. In each ACO run, both the overall makespan (F) and the device makespans (F^{RL} andF_j^V) are recorded and compared with the DD results. The ACO search patterns (in terms of best results found as well as the dynamics of the Pheromone matrix along the ACO iterations) are studied to make sure the ACO algorithm is working properly. By studying the ACO's search patterns, important insights are obtained regarding the scheduling problem: in some problems the best F is found when either $F^{RL} > BestRL$ or $max\left(F_j^V\right) > BestVmax$ (thus the best overall makespan does not necessarily have the best shuttle/retrieval makespans), which indicates the complex interaction between shuttle scheduling and retrieval lift scheduling. To further secure the optimization qualities of the ACO results, five independent ACO replications are performed for each task set, and the best results among replications are selected. We observed that the replication error ranges are very small for all problems (typically within $\pm 0.2\%$ of the replication averages), indicating that the ACO has explored the solution space adequately for each problem. The tests are conducted on the same 2.60GHz clock speed, 8GB RAM computer for testing the previous problem. Each of the 40 problems takes about 200,000 iterations (roughly one hour) to solve, and the best solution is found within 50,000 iterations (roughly 15 minutes) on average. The ACO and DD results are illustrated in Figure 5.14. The lower bounds of each problem size determined as $\max(2S\omega^{TL}, 2R\omega^{TL})$ are also displayed.

It is observed that all best ACO results found are at least as good as the DD results (in term of both the overall makespan F and the device makespans F^{RL} and $max(F_j^V)$) in all 40 problems. In most problems, better results are found by ACO than by the DD approaches – these better results are usually found after larger numbers of ACO iterations, while the gaps between ACO and DD results are very small for all makespan indicators. For the two DD approaches, the average gaps (in term of F) of A*-Closest and A*-FCFS results to ACO results are both 0.9%, and no significant difference is observed between the makespan results from the DD approaches in each problem. In addition, the average gaps for F^{RL} and $max(F_j^V)$ between DD and ACO are also very small (1.2% and 0.9%, respectively). It is noticeable that the ACO results may not be "true optimal", not only due to the heuristic nature of this optimization algorithm, but also because the shuttle schedules are constructed using A* Search Algorithm (which is implemented differently from its implementation in the DD approaches as illustrated previously in Figure 5.12) in each solution.

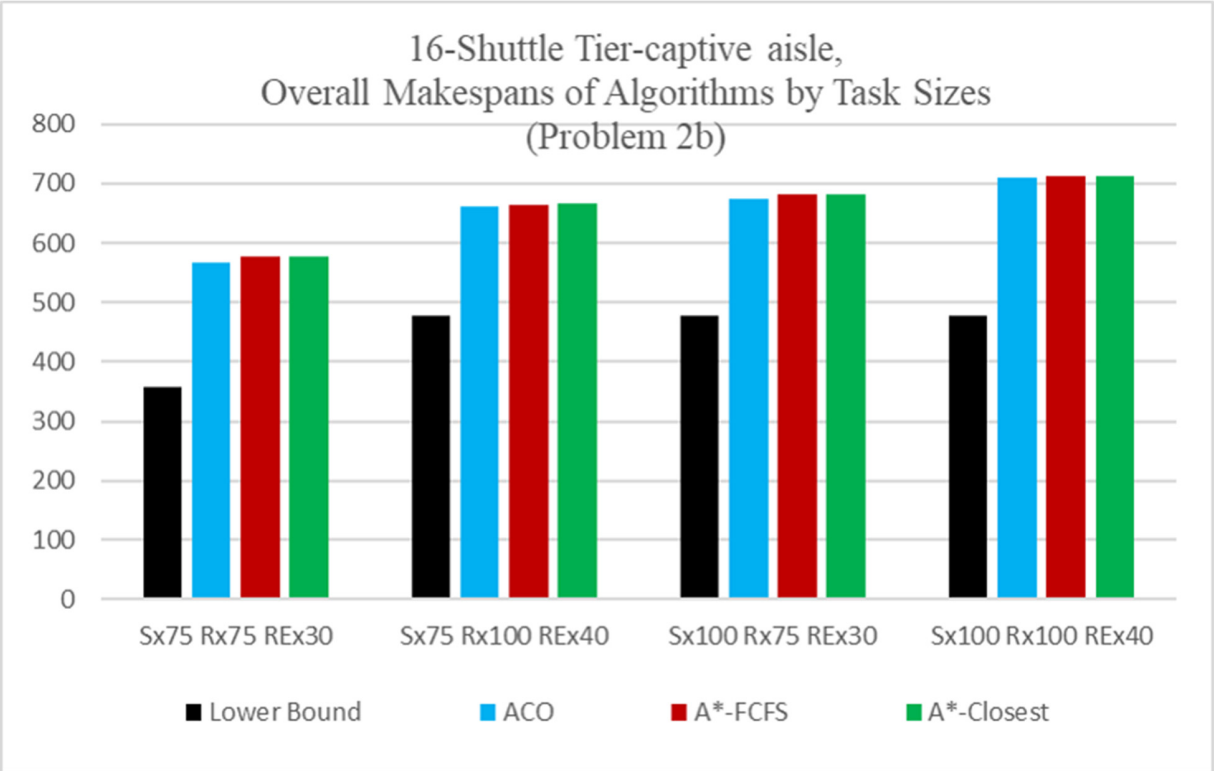

Figure 5.14 Dynamic Dispatching Results vs. Optimization Results (Problem 2b, algorithm averages of 10 problems for each task size)

But still, in many problems the ACO found better shuttle schedules than those found in the DD approaches – although the A* Search Algorithm in ACO is subject to additional consistency constraints with each ACO solution, schedules with better overall makespan might be found through heuristically exploring large numbers of feasible ACO solutions. Figure 5.15 illustrates the retrieval lift makespan F^{RL} and the maximum shuttle makespan $\max(F_j^V)$ from the best results found in each ACO iteration (logarithmic scale in the x-axis) for an ACO replication of a [100 S, 100 R, 40 Re] problem, and the results obtained from the A*-Closest approach are posted as horizontal lines for comparison. In this problem, $F^{RL} > \max(F_j^V)$ is observed in both A*-Closest results and ACO results, which means the storage tasks (whose final service stages are shuttle services) are completed earlier than the retrieval tasks (whose final service stages are retrieval lift services). This observation indicates the system is in general more burdened by the retrieval tasks (incl. the relocation workloads) rather than by the storage tasks (note that it is not always the case for different problems). It can also be observed that $max(F_j^V)$ in the best ACO results once got larger in later iterations, which indicates better F_j^V's not necessarily bring better overall makespan because there is generally $F^{RL} > \max(F_j^V)$ in this problem. As the figure illustrates, the ACO found results as good as those obtained by A*-Closest in a few hundred iterations (a few seconds on the test computer). However, it became increasingly difficult for the ACO to explore further improvements – the best solution is found at iteration 43,973 (about 15 minutes on the test computer) with only 0.9% improvement comparing to the A*-Closest solution. Then, the ACO could not find further improvements in the next 100,000 iterations (about 40 minutes) and get terminated. Similar ACO search patterns are found with other problems, and the gaps between DD and ACO results are mostly stable around the aforementioned averages.

As discussed above, average gap between the DD and ACO results is 0.9%. Considering the best results pattern of our ACO implementation (as previously illustrated by the example in Figure 5.15), it is reasonable to view the DD results as satisfactorily close to the "true optimal solution" which may not yet found by the ACO. Considering the simplifying assumptions made in this problem as well as in the previous problems (e.g. all tasks are known at the beginning, and all storage assignment decisions are already made), both of the proposed DD approaches that perform shuttle scheduling and retrieval lift scheduling are viewed as solid bases for further control strategy

development in dynamic environments (continuous task arrivals) and with the presence of storage assignment decisions and tier-transfer scheduling.

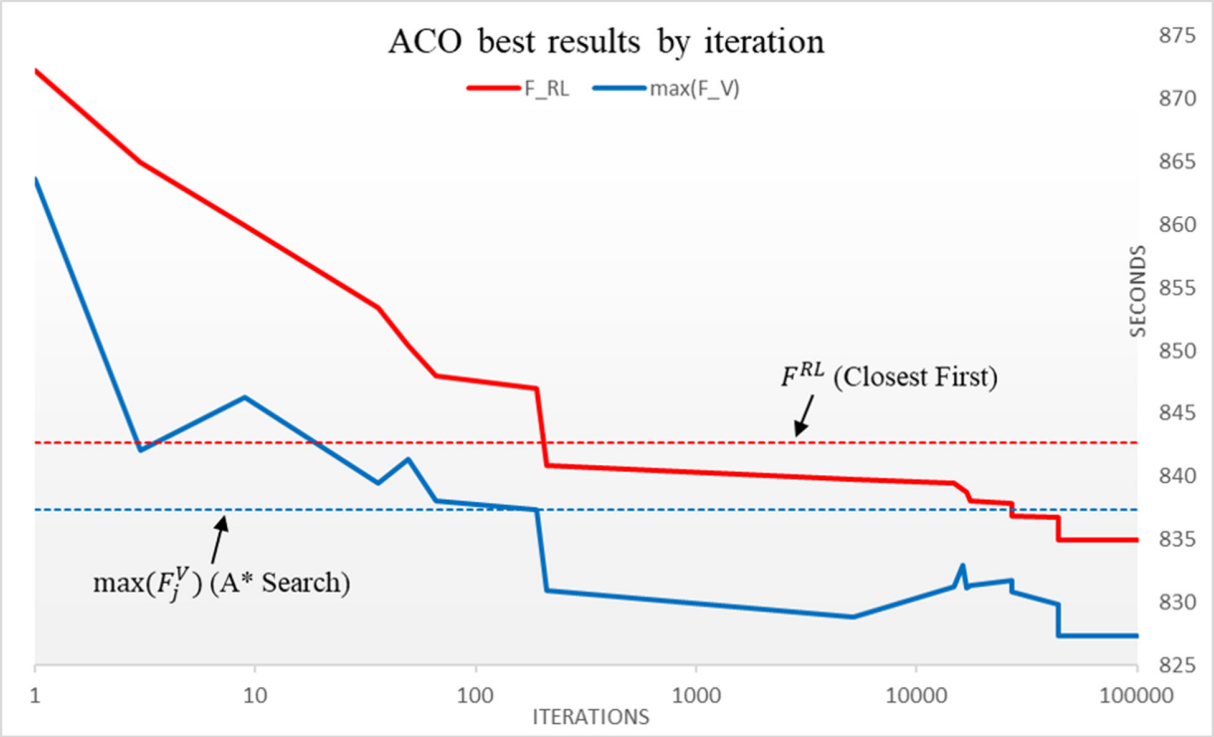

Figure 5.15 ACO best results by iteration of a [100 S, 100 R, 40 Re] problem

Problem 2c: Scheduling all devices in aisle for Tier-to-tier SBS/RS

In tier-to-tier configurations, shuttles are transferred between tiers by the shuttle lift through tier-transfer operations. The selected shuttle moves to the transfer station located at the opposite end of the aisle (from the storage and retrieval lifts) and then waits for the shuttle lift. Then, the shuttle is loaded by the lift, transported to the target tier, and then unloaded to the transfer station of the target tier. Tier-transfer services are time-costly, and the occurrence of tier-transfer operations should be minimized through control approaches. Figure 5.16 illustrates an example of devices scheduling for a deterministic task set in a 2-deep, 8-tier, 3-shuttle aisle. 8 storage tasks and 8 retrieval tasks (3 relocations) are targeting 7 of the 8 tiers, and 4 tier-transfer operations are performed. Note that one shuttle (V1) is transferred twice in this example (tier 1 to tier 3, and tier 3 to tier 6). The occurrence rate of tier-transfer operations (θ^T) is as high as $4/(8+8) = 25\%$ in this example because the task set is relatively small. It can be inferred that with a tier-to-tier aisle

of X tiers, J shuttles and given a deterministic task set in which X' tiers are involved, the minimum occurrences of tier-transfer operations are $(X' - J)$ – obviously, the tier-transfer occurrences are minimized if the shuttles are selected for tier-transfer only after there are no remaining tasks on the selected shuttles' current tiers. In addition, it can be inferred that the storage assignment policy is potentially more significant in Tier-to-tier configurations than in Tier-captive configurations. In the same example in Figure 5.16, the storage totes are arranged and assigned to different tiers in "good" locations so that the shuttles are never suspended waiting for the upstream storage lift services. Image if the storage tasks are served in the reverse sequence $(S_8, S_7 \ldots S_1)$ by the storage lift, the shuttle makespans on the shuttles' initial tiers will be longer due to less scheduling flexibility and the possible waiting times, as a result the overall makespan will also be worse.

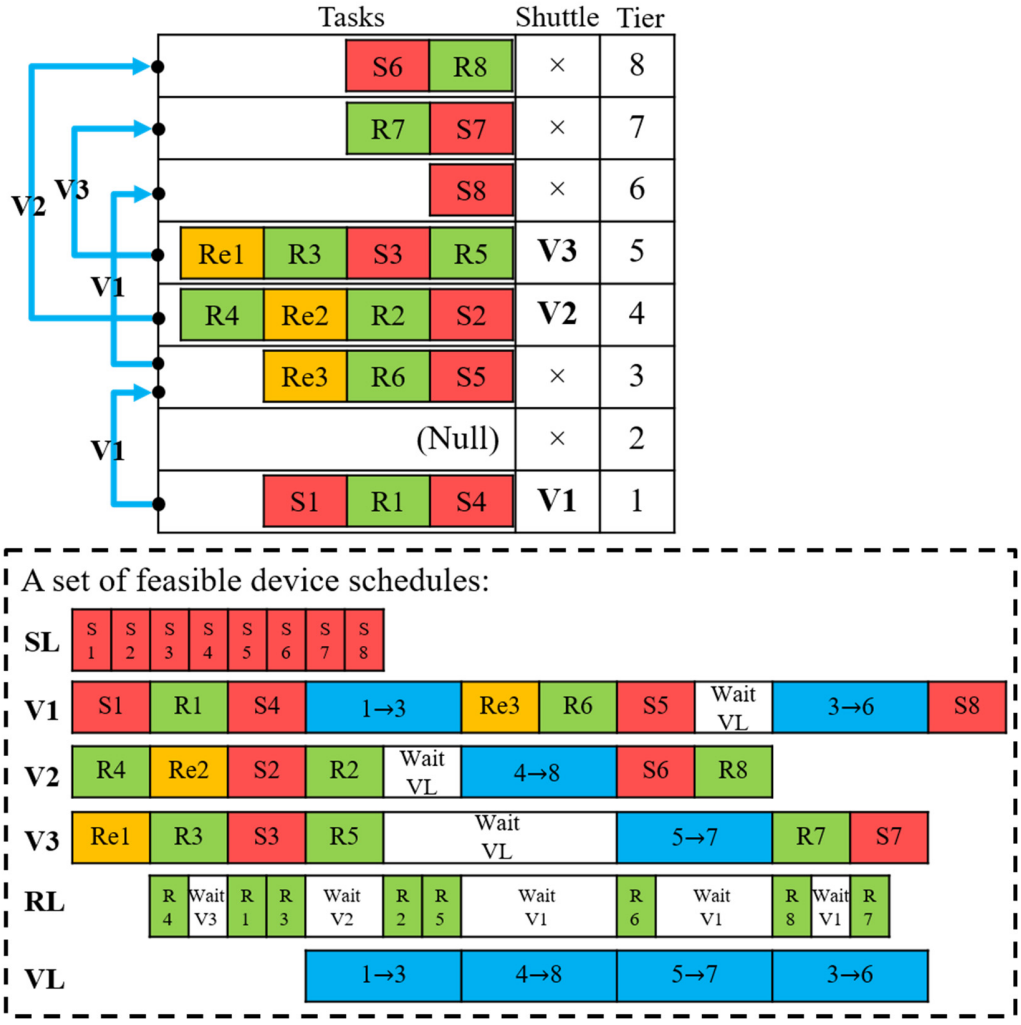

Figure 5.16 A devices scheduling example in a Tier-to-tier aisle

Because the tier-transfer operations are time costly, it is assumed that a shuttle will be selected for tier-transfer only after it has completed all the tasks on its current tier, thus the total number of tier-transfer operations equals $T = (X' - J)$. In some extreme cases with regard to the storage task arrivals, it is possible that doing more than T transfers will be better decisions for minimizing the overall makespan, while such extreme cases are not concerned for now as the storage assignment decisions are assumed to be well made. Note that in the current phase where MP approaches are applied to provide baselines for control strategy development, various simplification assumptions are made, in particular the task sets to the problems are assumed deterministic which is not the case when the control strategies are finally implemented for serving continuously arriving tasks in practical environments.

The tier-transfer assignment problem consists of two decision aspects. First, the target tier of each operation needs to be determined. Second, a shuttle is selected for each tier-transfer operation. In each of the T tier-transfer decisions, a single shuttle is selected from J shuttles in the aisle, and a single tier is selected from the remaining tiers that need a tier-transfer. When $J < T$, the tier-transfer assignment problem resembles a resource allocation problem subject to initial resource availabilities – J resources (shuttles) are allocated to T jobs (tiers), where each resource is not available until a specific time (when the shuttle completes all tasks on its initial tier). The resource allocation problem is also subject to setup times, where the "setup time" of each allocation is the sum of three elements: 1) the shuttle travel time from its end location after served its current tier to the transfer tier location; 2) the shuttle lift travel time from the tier of the last allocation to the shuttle's current tier; and 3) the total tier-transfer service time including loading the shuttle, travel time from the shuttle's current tier to the target tier of the current allocation, and unloading the shuttle. Obviously, the setup times of this resource allocation problem are sequence-dependent because the devices' end locations and thus travel times are subject to the previous tier-transfer operations. The total number of decision combinations is obtained as $JT \times J(T-1) \times J(T-2) \times \ldots \times J = J^T \times T!$ even if the scheduling details of the shuttle tasks on each tier are not considered in the decisions. The solution space of this problem could be extremely large for typical Tier-to-tier configurations (e.g., 8.036×10^{15} for $X = 16, J = 4, T = 12$). However, this problem can be reasonably simplified in the form of a single-resource scheduling problem where T tiers are to be visited by tier-transfer operations. Each feasible solution can be presented as a size-T

array $[u_1, u_2 \dots u_T]$ where u_i indicates the tier index of the ith transfer, and the solution space size is thus reduced to $T!$. The simplification assumptions are stated as follows:

1) When a shuttle is available to each tier, the shuttle service sequence for that tier's tasks is assumed to follow a predefined scheduling rule. Under this assumption, given the available time and the initial location of the shuttle, the schedule and makespan of shuttle tasks on each tier can be estimated regardless of which shuttle is selected for tier-transfer operations. The predefined rule here is determined as A* Search Algorithm as it is viewed as near-optimal in the previous problems.
2) The storage lift serves tasks in FCFS and the storage locations are predefined. The storage assignment decisions are assumed to be "good" so that the makespan estimates based on assumption 1 are acceptable (recall the observations from Problem 1d that the makespan is subject to task time windows). Also, good storage assignment decisions indicates that the shuttle waiting times for storage task arrivals are in general smaller than the tier-transfer service times (otherwise, the best solutions might be obtained by performing more than T transfers, but the overall control approach is poor because such cases should be minimized from the storage assignment side).
3) The target tier of the next tier-transfer service is selected according to the sequence defined in the solution, but the shuttle for this service is selected from all shuttles according to a predefined rule. Under this assumption, the shuttle for each tier-transfer service is deterministic, and the shuttle arrival times (available times) and initial locations for each tier's shuttle tasks are both known. Then, each shuttle's schedules on all its allocated tiers, as well as its total can be estimated under assumptions 1) and 2), and its total makespan F_j^V (incl. tier-transfer travel and waiting times) is obtained. The predefined rule here is determined as follows: given the shuttle lift's current tier x_0 and the next transfer's target tier x, select the shuttle $j \in \{1 \dots J\}$ currently on tier x_j which has the minimum value of $D_{x_0, x_j, x}$ estimated as the sum of waiting time and service time for this tier-transfer service.
4) The retrieval lift schedule is assumed to follow a predefined rule deterministic if given the shuttle schedules, and its makespan F^{RL} is obtained. This assumption is consistent with the previous problem 2b, and the overall makespan is also obtained as $F = \max\left[F^{RL}, max\left(F_j^V\right)\right]$. In the previous problem, both FCFS and Closest-First are viewed as near-optimal for retrieval

lift scheduling (when integrated with A* Search Algorithm on the shuttles side). The FCFS rule is determined as the predefined retrieval scheduling rule here.

Problem Inputs:

T : The number of transfer-needed tiers (equals the number of tier-transfer operations), there is $T = X' - J$;

$\boldsymbol{X^I}$: Set of tiers with initial shuttle availabilities, there is $|\boldsymbol{X^I}| = J$;

$\boldsymbol{X^T}$: Set of tiers without initial shuttle availabilities, there is $|\boldsymbol{X^T}| = T$ and $\boldsymbol{X^I} \cap \boldsymbol{X^T} = \emptyset$;

$t0^{VL}$: The earliest retrieval lift available time;

$x0^{VL}$: The initial tier of the retrieval lift when it becomes available;

$t0_x^V$: The earliest shuttle available time for transfer tier after it has completed the tasks on tier $x \in \boldsymbol{X^I}$ according to A* Search Algorithm;

$y0_x^V$: The initial column of the shuttle on tier $x \in \boldsymbol{X^I}$ when the shuttle becomes available;

M_x : The estimated service time of the tasks on tier $x \in \boldsymbol{X^I} \cup \boldsymbol{X^T}$ from the time when a shuttle is available on this tier. Denote function $f_{A*}^V(x, y_0)$ as the estimated makespan for tier x's task set according to the A* Search Algorithm and assuming a shuttle is initially available at column y_0. If $x \in \boldsymbol{X^T}$, the makespan is estimated by temporarily assuming all storage tasks to this tier are already delivered to the input buffer. There are:

$$M_x = \begin{cases} f_{A*}^V(x, y0_x)\,, if\ x \in \boldsymbol{X^I} \\ f_{A*}^V(x, Y)\,,\ \ if\ x \in \boldsymbol{X^T} \end{cases}$$

Y_x : The shuttle's end location on tier $x \in \boldsymbol{X^I} \cup \boldsymbol{X^T}$ after completing this tier's task set according to A* Search Algorithm;

$D_{x_0, x_j, x}$: The tier-transfer costs from tier x_j to x when the shuttle lift is at tier x_0, consists of three components: 1) the shuttle travel time from its end location after completing its current tier x_j's tasks to the transfer tier location, 2) the shuttle lift travel time from its end location x_0 after its previous tier-transfer service to the shuttle's current tier x_j, and 3) the total tier-transfer

service time including loading the shuttle, travel time from the shuttle's current tier to the target tier x_2, and unloading the shuttle. Computed as follows:

$$D_{x_0,\,x_j,x} = \tau^{V}_{Y_{x_j},Y} + \tau^{VL}_{x_0,\,x_j} + \left(\tau^{VL}_{x_j,x} + 2\omega^{VL}\right)$$

Where $\tau^{V}_{Y_{x_j},Y}$ is the shuttle travel time from its end location Y_{x_j} to the transfer station, $\tau^{VL}_{x_0,\,x_j}$ and $\tau^{VL}_{x_j,x}$ are shuttle lift travel times, and ω^{VL} is the load/unload time of the shuttle lift.

Solution Coding:

A size-T array $[u_1, u_2 \dots u_T]$ where $u_i \in \boldsymbol{X^T}$ indicates the tier index of the ith transfer.

Search Method:

Based on the previous assumptions, the objective function is formulated as $F = \max\left[F^{RL}, max\left(F_j^V\right)\right]$, where F^{RL} is the retrieval lift makespan and F_j^V is total makespan of shuttle $j \in \{1 \dots J\}$. The total size of the solution space equals $T!$ – because it is not a very large number with typical Tier-to-tier configurations (e.g. for a 16 tiers aisle, $T! = 4.79 \times 10^8$ with 4 shuttles, 40,320 with 8 shuttles, and 24 with 12 shuttles: assuming all tiers are involved in the tasks), an enumeration search algorithm is applied to find the best solutions. A simple Discrete Event Simulation (DES) model is integrated in the search algorithm to return the makespan of each solution, described in pseudocodes as follows:

$Solution = [u_1, u_2 \dots u_T]$

/* location and time when each shuttle finished serving tasks on its current tier */

$x_j^V = \boldsymbol{X^I}(j),\ \ y_j^V = Y_{\boldsymbol{X^I}(j)},\ \ t_j^V = t0^V_{\boldsymbol{X^I}(j)} + M_{\boldsymbol{X^I}(j)},\ \ \forall j \in \{1,2 \dots J\}$

/* location and time when the shuttle lift finished its current tier-transfer service */

$x^{VL} = x0^{VL},\ \ t^{VL} = t0^{VL}$

FOR $(1 \le i \le T)$

/* earliest time to start the next tier-transfer service*/

$time = \max\left[min\left(t_j^V\right), t^{VL}\right]$

FOR $(1 \le j \le J)$

$f(j) = \max\left(t_j^V - time, 0\right) + D_{x^{VL},\,x_j^V,\,\boldsymbol{X^T}(u_i)}$

END FOR

$Selected = j|\min[f(j)]$

/* update device states */

$x^{VL} = \boldsymbol{X^T}(u_i),\ t^{VL} = time + f(Selected)$

$x^V_{Selected} = \boldsymbol{X^T}(u_i),\ y^V_{Selected} = Y_{\boldsymbol{X^T}(u_i)},\ t^V_{Selected} = t^{VL} + M_{\boldsymbol{X^T}(u_i)}$

END FOR

$F^V_j = t^V_j, \forall j$

Compute F^{RL} based on the resulting shuttle schedules according to FCFS

Return Makespan: $F = \max\left[F^{RL}, max\left(F^V_j\right)\right]$

Evaluation:

A 2-deep, 16-tier, 200-column aisle design is determined as the system for testing this problem. Five shuttle configurations are selected for testing: 4, 6, 8, 10 and 12 shuttles, and 10 task sets with [100 S, 100 R, 40 Re] are generated randomly with equal probabilities to each tier. Thus, totally 50 test problems are created. Because the enumeration approach explores the full solution space of each problem, mean, max, average, and standard deviation statistics of the solution spaces are also recorded, denoted as MIN(SS), MAX(SS), AVG(SS) and STD(SS), respectively.

Four Dynamic Dispatching rules are determined as evaluation candidates:

1) *MaxNT-Closest*: Select the next tier with the largest number of tasks (incl. relocations), then select the shuttle with the smallest travel costs $f(j) = \max\left(t^V_j - time, 0\right) + D_{x_0, x_j, x}, j \in \{1 \dots J\}$. Note that the first part of the equation is the time waiting for the shuttle to complete serving its current tier, and the second part is the total tier-transfer service time.
2) *MaxMS-Closest*: Select the next tier with the largest estimated makespan M_x, then select the shuttle with the smallest travel costs $f(j) = \max\left(t^V_j - time, 0\right) + D_{x_0, x_j, x}, j \in \{1 \dots J\}$.
3) $P||C_{max}$*-Closest*: Pre-allocate tiers to shuttles. Denote set $\boldsymbol{T^V_j}$ as the transfer-needed tiers allocated to shuttle $j \in \{1 \dots J\}$. The total workload (excl. tier-transfer costs) of shuttle j is estimated as $C_j = \left[t0^V_{\boldsymbol{X^I}(j)} + M_{\boldsymbol{X^I}(j)}\right] + \sum_{x \in \boldsymbol{T^V_j}} M_x$, where the first part of the equation is the shuttle's makespan on its initial tier $\boldsymbol{X^I}(j)$. The allocation is performed by temporarily viewing

the problem as a $P||C_{max}$ problem. According to definition by Graham et al. (1979), a $P||C_{max}$ problem is described as follows: given *n* jobs each characterized by a processing time p_j (j = 1, . . . , *n*), and *m* identical parallel processors each of which can process at most one job at a time, assigning each job to a processor so that the maximum completion time of a job (makespan) is minimized. In our approach, $\boldsymbol{T_j^V}$'s are developed using a Move-Swap local search heuristics trying to minimize the maximum shuttle workload $max(C_j), j \in \{1 \dots J\}$ – this heuristic approach will be introduced in detail in Section 5.5. Then, in each tier-transfer assignment decision when the shuttle lift is currently at tier x_0, the next tier to be served is determined as the one has the smallest time costs $f(x) = \max(t_j^V - time, 0) + D_{x_0, x_j, x}$, where $j \in \{1 \dots J\}$ and $x \in \boldsymbol{T_j^V}$.

4) $P||C_{max}$-A*: An A* Search Algorithm based on $P||C_{max}$ approach. The main idea of this approach is very similar to the A* Search Algorithms applied in the single-shuttle scheduling problems studied earlier. Tiers are pre-allocated to shuttles in the same heuristic as in the previous $P||C_{max}$-Closest approach. Then, in each tier-transfer assignment decision when the shuttle lift is currently at tier x_0, the next tier is determined as $x^* = \min[g(x) + h(x)] \mid x \in \{remaining\ tiers\}$, where $g(x) = \max(t_j^V - time, 0) + D_{x_0, x_j, x}$ (same as $f(x)$ in the $P||C_{max}$ -Closest policy), and $h(x)$ is the estimated overall makespan by temporarily assuming $P||C_{max}$-Closest will be applied for all the remaining tiers.

The statistics of the solution spaces as well as the dynamic dispatching results are illustrated in Figure 5.17. The problem averages of the solution spaces' MIN, MAX, AVG and STD are presented. The MIN(SS)'s are viewed as the baselines for the evaluation. Also, it is reasonable to view the AVG(SS) results as indicators of random solution performances when no prioritization for tiers is applied (but still, a greedy-search method is applied for shuttle selection as described previously in the search method pseudocode) – the overall gap between MIN(SS) and AVG(SS) is 13.1%. Two DD policies appear to be more advantageous than the other two: the MaxMS-Closest and the $P||C_{max}$-A*, and their overall gaps to MIN(SS) are 2.4% and 1.3%, respectively, and both cost less than 10 milliseconds computational time in all problems. It is further observed that with the $J > T$ configurations ($J = 10, 12$ and $T = 6, 4$), the MaxMS-Closest policy found the optimal solutions (gaps to MIN(SS) are 0 for all problems) – this is reasonable because in such cases, this greedy method which always assigns the shuttle with the

smallest travel costs to the most burdened tier will generally guarantee the minimum overall makespan. However, with $J = T = 8$, P||C_{max}-A* (average gap to MIN(SS) = 0.6%) slightly outperforms MaxMS-Closest (average gap = 1.2%). Then, with $J < T$ cases configurations ($J = 4, 6$ and $T = 12, 10$), the performance of MaxMS-Closest deteriorates to 3.6% and 7.3% in terms of average gaps, while for P||C_{max}-A* the numbers are 2.0% and 2.7%. Such observations indicate that control strategies may need to be customized for tier-to-tier configurations according to the aisles' shuttle fill rates. At this point, with control strategy development for more complex dynamic operations under practical environments (as opposite to the MP analysis in this section where task sets are assumed deterministic), we view the MaxMS-Closest policy as promising for $J > X - J$ configurations and view the P||C_{max}-A* policy as a promising for $J \leq X - J$ configurations.

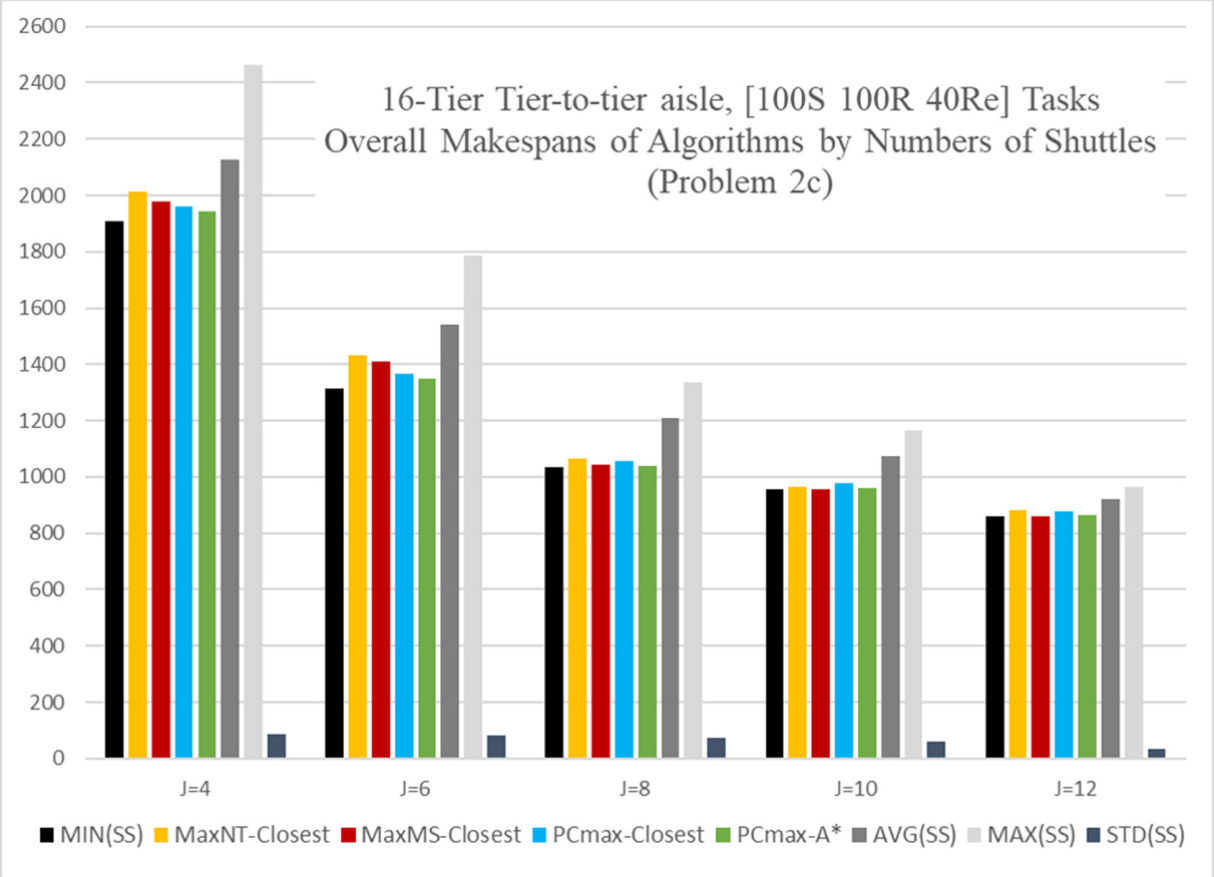

Figure 5.17 Dynamic Dispatching Results vs. Solution Space Statistics (Problem 2c, algorithm averages of 10 problems for each shuttle deployment quantity)

5.5 Development and Analysis of Storage Assignment and Relocation Algorithms

The development of storage assignment and relocation approaches needs to consider both short-term and long-term effects to the system performance, described as follows:

1) Short-term effects: the instantaneous effects to current tasks (e.g., increment to the expected overall makespan of device schedules) if specific assignment decision(s) are made for the current storage task(s); and
2) Long-term effects: the expected system performance regarding future storage and retrieval task arrivals (e.g., expected throughput or task cycle times) if assignment decisions are made according to specific rules.

As studied in Chapter 4, the system performances is dependent on the utilization of the S/R devices, which is primarily determined by the task arrival rates and rack utilizations. Generally speaking, the short-term effects are more important when the device utilizations are relatively low – particularly in manufacturing environments where responsiveness requirements are posed by time-constrained processing orders. On the other hand, in application environments like E-commerce distribution centers, the long-term effects are more important to achieve better sustainable system performances like overall throughputs, and the tasks are usually less time-sensitive in such environments.

Just like in the previous section where device scheduling problems are studied, we attempt to analyze the storage assignment/relocation problems in increasing complexities regarding system designs and demand assumptions. On the other hand, compared to the device scheduling decisions, storage assignment/relocation decisions are more subject to demand scenario assumptions, including but not limited to S/R arrival patterns, rack utilization, SKU-level characteristics (turn-over rates, affinities, etc.). Thus, unlike the mathematical formulation and benchmarking approaches conducted for the scheduling problems, in this section the focus is primarily on providing general insights and high-level procedures for developing the storage assignment/relocation policies.

5.5.1 Storage Assignment and Relocation without explicit SKU-level Characteristics

This subsection studies storage assignment and relocation decision-making in demand scenarios where SKU-level characteristics are not explicitly known to the decision-maker – for example, the SKU turnover rates, SKU affinities (correlations in orders), as well as order time constraints (due-date, lateness, etc.) are not considered in the storage assignment decisions. Thus, totes stored in the racks have equal time-dependent probabilities to be retrieved in the future.

Closest Open Location (COL)-type policies are the primary focus in the development storage assignment/relocation approaches. In traditional crane-based AS/RSs, the COL policy can potentially provide optimal throughput performance under the assumptions for implicit SKU-level characteristics ("potentially" because other factors like dual-cycle operations, device dwell location control, and multi-capacity crane scheduling, etc. still need to be considered). This is because in each aisle of such AS/RSs, the crane is the only S/R device responsible for both horizontal and vertical movements, and all the slots can be prioritized by deterministic travel times from/to I/O for COL implementation. As discussed in the previous chapters, such prioritization is more complicated in SBS/RS due to the service patterns of heterogonous, interacting S/R devices.

Tier-captive Aisles

In tier-captive 1-deep aisles, each tier has an identical, captive shuttle, the expected service and waiting times on the shuttles' sides for both task types ($T_S^V, W_S^V, T_R^V, W_R^V$) are minimized from the storage assignment perspective by balancing the shuttle workloads. Call this approach *horizontal-focused COL*, it is achieved through two steps:

1) Assign the next storage tote to the tier with the most available slots; and

2) Apply COL policy that selects the closest available slot to the I/O buffers on the selected tier.

On the other hand, the service performance on the tote lifts (both types) sides are dependent on the storage or retrieval arrival rates to each tier – under the SKU assumptions here, those task arrival rates are proportional to the rack utilization on different tiers. Storing more totes to the lower tiers is expected to reduce the lifts' service/waiting times ($T^{SL}, W^{SL}, T^{RL}, W^{RL}$). With multi-capacity tote lifts cases, the retrieval lift service times can be further reduced through scheduling

approaches (as studied in Problem 2a, Section 5.4). As for the storage lifts, although the sequence of storage totes is assumed inflexible due to the roller-conveyor characteristics of the upstream system, the service/waiting times (T^{SL}, W^{SL}) can be reduced through storage assignment decisions. For example, when the tasks in the storage lift's queue is larger than one, assigning slots on the same tier in the same tour will reduce the service times. Call the above approach *vertical-focused COL*, the following procedure are performed periodically (e.g., hourly) in the routine system operations:

1. Initialize array $[\rho_1, \rho_2 \dots \rho_x \dots \rho_X]$ where $\rho_1 = \rho_2 \dots \rho_x \dots = \rho_X = \rho$;
2. Use array for slot task visit probability estimation in the analytical model (introduced in Chapter 4), estimate system performance;
3. Update array so that it satisfies $0 < \rho_1 < \rho_2 \dots \rho_x \dots < \rho_X < 1$ and $\sum \rho_x = X\rho$;
4. Use array for slot task visit probability estimation in the analytical model, estimate system performance;
5. Compare performance estimates, go back to 3 or stop (return array).

Then, vertical-focused COL is described through modifying step 1 of horizontal-focused COL as follows:

1) Assign the next storage tote to tier $x = \max\{(\rho_x YZ - N_x) : x = 1 \dots X\}$, where N_x is the number of currently occupied slots on tier x;

The vertical-focused COL will lead to larger service times on the shuttles' sides because the tier/shuttle workloads are less balanced. Theoretically, it is preferable for system designs where the tote lifts are the bottleneck rather than the shuttles. According to the simulation experiment results in Chapter 4 using the parameters provided by the sponsor, we illustrated in Figure 5.18 the ratios of shuttle utilizations (U^V) and tote lift utilizations (U^{TL}) as measures of bottlenecks in different designs (thus the shuttles are the bottleneck if $U^V/U^{TL} > 1$). We observed that both the shuttles and the tote lifts can become the bottleneck, while the shuttles are more constraining the system performances when rack aspect ratios (Y/X) get larger – which is consistent with general intuition.

However, unlike the shuttles, the tote lifts' travel times are usually relatively small portions of their service times. When the SKUs are assumed implicit, we found in simulation experiments

that unbalancing tier workloads with vertical-focused COL is not very effective in improving the system performance, especially when the overall rack utilization of the aisle is large (e.g., $\rho > 0.8$). Thus, the horizontal-focused COL approach is viewed as preferable for most tier-captive designs with implicit SKU-level characteristics. On the other hand, the general procedure of the vertical-focused COL approach provides a basis for storage assignment policies when SKUs are explicit.

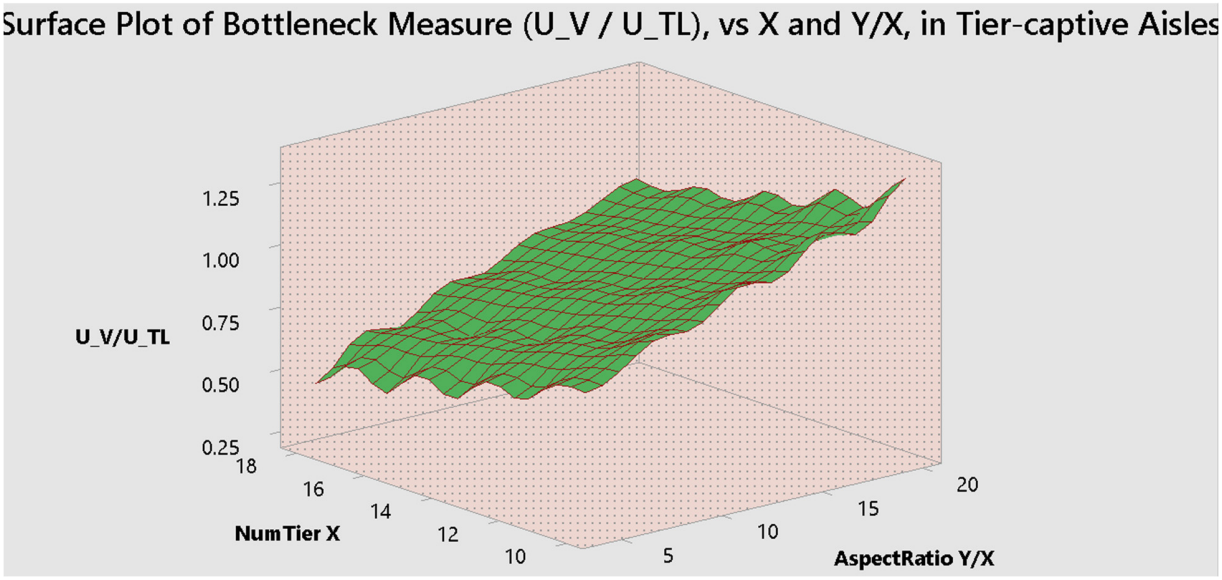

Figure 5.18 Tier-captive aisles' bottleneck patterns by designs, measured as dividing shuttle utilization by tote lift utilization: U^V/U^{TL}, based on simulation experiments that the aisle is in steady-state with $\max(U^V, U^{TL}) = 0.9$

Tier-to-tier Aisles

In tier-to-tier SBS/RS, the number of shuttles is less than the number of tiers in each aisle, thus shuttles are transferred between tiers by the shuttle lifts when necessary. Because the shuttles are not captive to the tiers, the decision maker has additional options here in using the tiers according to the rack utilization dynamics. Figure 5.19 illustrates four general types of storage assignment options in a tier-to-tier aisle:

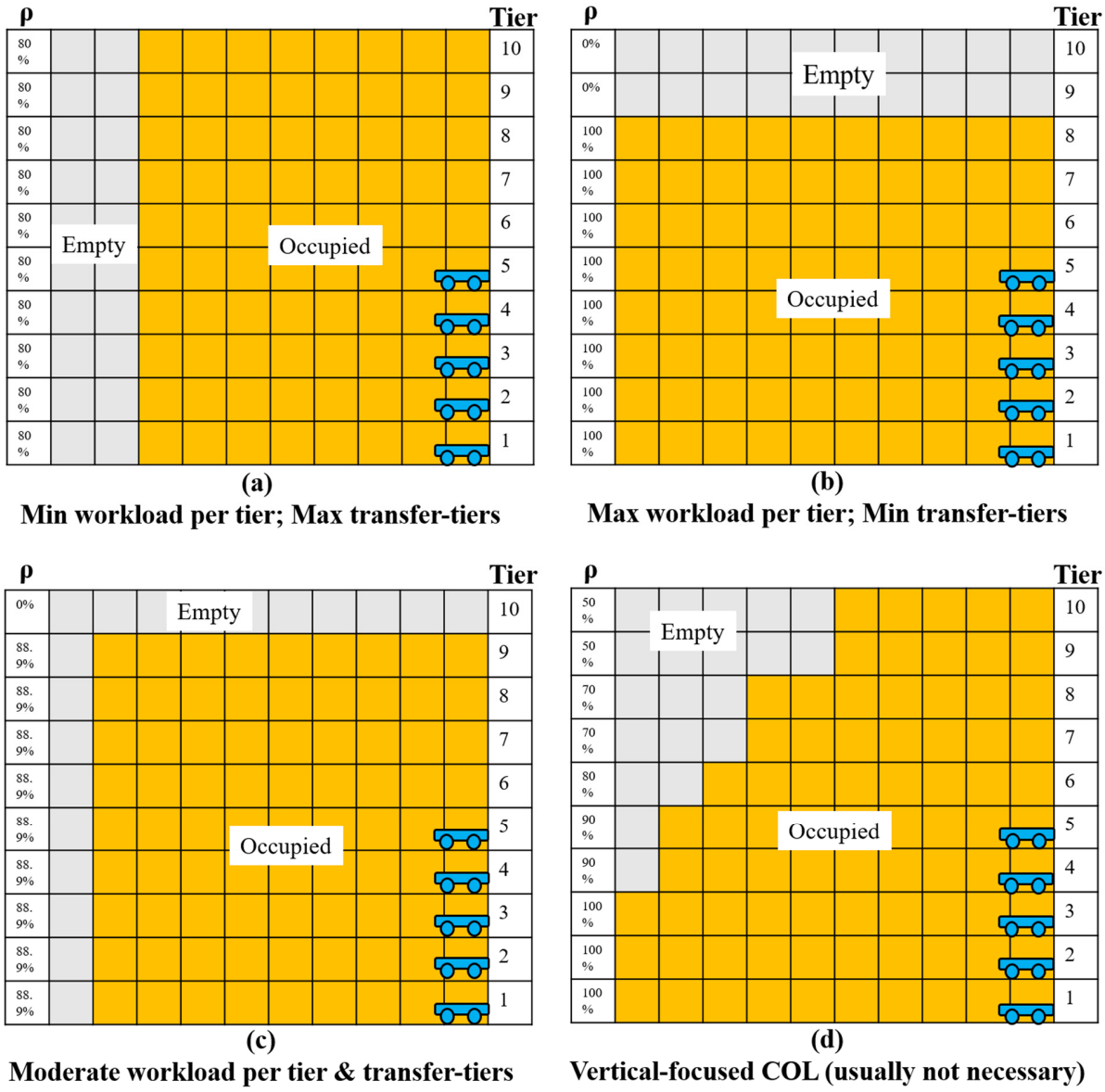

Figure 5.19 Storage assignment options for Tier-to-tier systems

a) Store totes uniformly to all tiers: the average workloads on each tier are minimized, while the relocation workloads are expected to be maximized. In the example, 5 shuttles are serving a 10-tier aisle where $\rho_x = 0.8$;

b) Store totes to lower tiers as much as possible: the relocation workloads are expected to be minimized, while the workloads on the utilized (lower) tiers are maximized. In the example, the system can be approximated as 5 shuttles serving an 8-tier aisle where $\rho_x = 1.0$;

c) Store totes uniformly to lower tiers: balance the in-tier workloads and tier-transfer workloads so that to minimize the overall service times. In the example, the system can be approximated as 5 shuttles serving a 9-tier aisle where $\rho_x = 0.889$;

d) Vertical-focused COL.

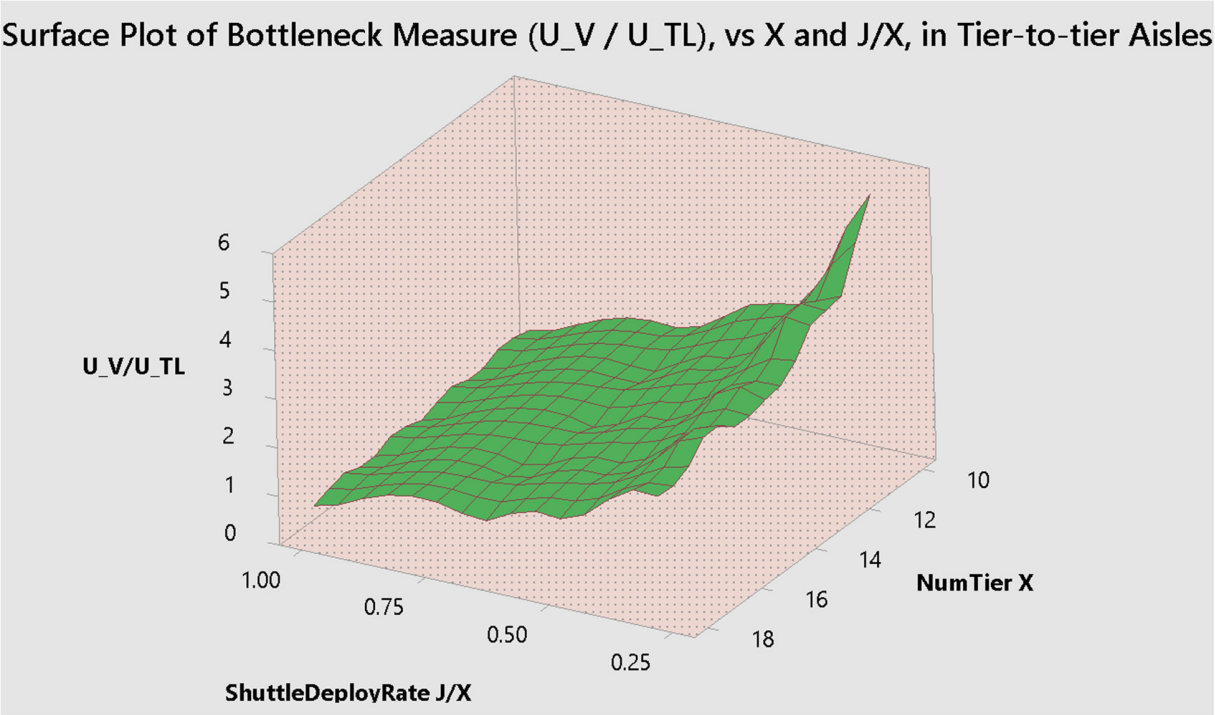

Figure 5.20 Tier-to-tier aisles' bottleneck patterns by design configurations, measured as dividing shuttle utilization by tote lift utilization: U^V/U^{TL}, based on simulation experiments that the aisle is in steady-state with $\max(U^V, U^{TL}) = 0.9$

When overall rack utilization is high (e.g., $\rho > 0.9$), options a), b) and c) are approximately identical. When ρ is expected to stay low in a sufficiently long duration (e.g., several days), options b) and c) can usually provide better system performances than option a). The decision maker can select the option whichever have better performance estimates using the analytical model proposed in Chapter 4, and always flexible to switch options when the expectation for ρ changes. Vertical-focused-COL (option d)) introduced previously for tier-captive systems is usually not necessary for tier-to-tier systems. According to the analytical approaches and simulation experiments from the previous chapters, the shuttles are always the bottlenecks in tier-to-tier systems (as illustrated in Figure 5.20, when $J/X < 1$ there is $U^V > U^{TL}$ in most design configurations) – thus reducing

lift travel times is a minor concern here in storage assignment decisions. This is also consistent with the common practice: the numbers of shuttles are configurable to accommodate the seasoning demands, but the tote lifts are fixed and expected to support the full-scale operations (when all tiers are deployed with shuttles).

2-deep racks

In aisles with 2-deep racks (either tier-captive or tier-to-tier), relocation services are performed by the shuttles. As analyzed in Chapter 4 and earlier this chapter, relocation services are usually as time-costly as storage or retrieval tasks, thus they should be minimized through storage assignment decisions. On the other hand, as discussed in the previous chapters, relocation decisions are similar to storage assignment decisions and follow the same rules. As illustrated in Figure 5.21, some general criteria should be followed in storage assignment and relocation decision makings in 2-deep aisles:

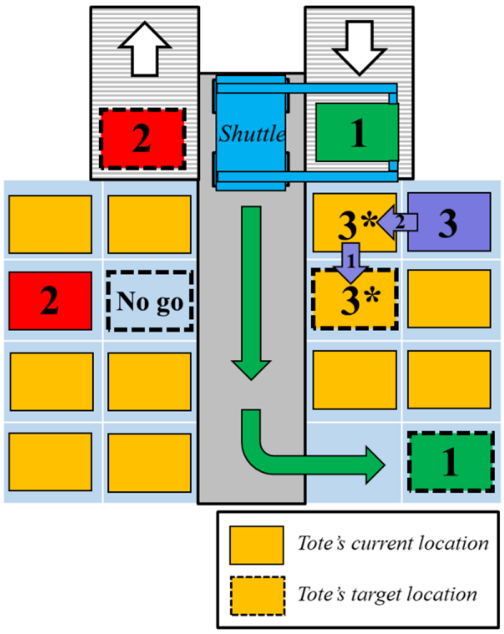

Figure 5.21 General storage/relocation criteria with 2-deep SBS/RS

1) Storage/relocation should not be assigned with any 1-deep slot who's neighboring 2-deep slot is empty (tote.1);
2) Storage/relocation should not be assigned with any 1-deep slot that will block existing retrieval tasks from its neighboring 2-deep slot waiting to be served by the shuttles (tote.2);

3) When a retrieval is blocked (tote.3), relocation of the blocker (tote.3*) should only be assigned to a slot within the tier of the corresponding retrieval task.

The first step of storage assignment decisions here (selecting the target tier) is the same as the step 1 in tier-captive 1-deep aisles (either horizontal or vertical-focused), while the second step (selecting the slot) is more complicated due to concerns for further relocations. In Chapter 4, when COL policy is assumed, the slot selection step is based on predetermined prioritization orders – all the 2-deep slots are prioritized than any 1-deep slots to minimize relocation occurrences, and then 1-deep and 2-deep slots are prioritized separately by the distances to the I/O buffers. A question that comes up is whether in racks of long horizontal distances (more columns), minimizing relocation guarantees minimization of the overall shuttle service times – considering the longer shuttle travel times, selecting 1-deep slots closer to I/O might be less time costly than selecting 2-deep slots at the far end of the aisle even when the future relocation costs from those 1-deep slots are taken into consideration. If such situation applies, there must exist a critical column $y^* \in [1, Y]$ that makes 1-deep slot $[x, y^*, z]$ at column y^* become preferable in storage assignment decisions than 2-deep slot $[x, 1, z+2]$ at column 1, $z \in [1,2]$. The two COL options for 2-deep rack storage assignment and relocation decision making are illustrated in Figure 5.22 – in option 2, 1-deep slots 15/16 closer to I/O will be preferable over 2-deep slots 17/18 at the far end if the additional travel time is larger than the expected relocation time costs (that will occur in the future when slot 1/2 are retrieved). If a critical column y^* does exist for a particular rack design, it can be approximated as follows:

$$y^* = \min(y) \mid \left[\left(\tau^V_{0,y} - \tau^V_{0,1}\right) > \left(2\overline{\rho}\tau^V_{0,Y/2} + 2\overline{\omega^V}\right)\right], y \in [1, Y]$$

Where $\tau^V_{0,y}$ is the shuttle travel time from I/O to column y, and $\overline{\omega^V}$ is the average L/U time in relocation services, and $\overline{\rho}$ is the expected rack utilization. Considering the shuttles' acceleration/deceleration patterns, there is always $2\tau^V_{0,Y/2} > \tau^V_{0,Y} > \tau^V_{0,y}, \forall y$. Thus, it can be inferred that, when $\overline{\rho}$ is large, COL option 1 is usually more effective in the long-term because the critical column y^* either does not exist or is found on the very far end which does not make COL option 2 significantly preferable. Hence, option 1 is considered as the primary storage/relocation rule for evaluating slots in the same tiers. However, the observations here are viewed as meaningful for future research especially when SKU-level characteristics like turn-over rates are considered.

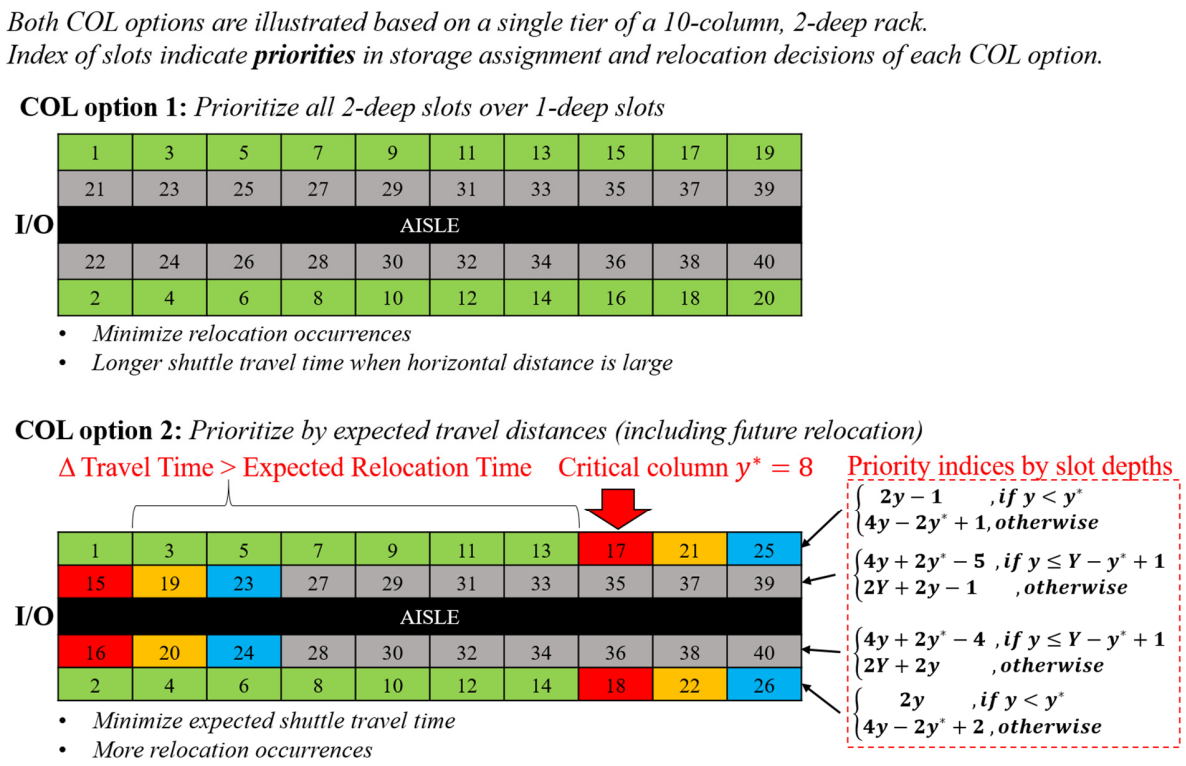

Figure 5.22 Two COL options for storage assignment & relocation in 2-deep racks

5.5.2 Storage Assignment and Relocation with Considerations for SKU-level Characteristics

SKU-level characteristics refer to product demand patterns which can be described as turn-over rates (popularity), demand correlations/dependencies (affinity), and demand trends (seasonality). The operational control of the system primarily focuses on the former two aspects. The popularity metrics are usually measured by the Cube per Order Index (COI) defined as the ratio between the space requirements and demand of a product (Goetschalckx and Ratliff, 1990), and popular products are expected to be assigned to more desirable storage locations that minimize the S/R service times. The affinity metrics represent how often two products are requested together in the same order, within the same time window or within a Bill of Materials (BOM) (Kofler, 2014). Data mining techniques that extract rules or patterns from the Warehouse Management System (WMS) database are often applied in the recent years to explore such SKU-level characteristics (Han et al, 2011).

As discussed in the previous section, the storage assignment decisions need to consider both the system designs/configurations and the devices' service patterns in SBS/RS. Particularly, the preference or desirability of each slot is determined by multiple factors including rack shapes, device speed/acceleration performances, and number of shuttles deployed. In general, we consider the following storage assignment/relocation criteria when SKU-level characteristics are explicit to decision-making:

1) Assign popular SKUs to preferable locations in the rack, which are slots closer to the I/O buffers to reduce shuttle service times, or/and slots on lower tiers to reduce lift service times;
2) In 2-deep aisles, assign SKUs of the same types in neighboring 1-deep and 2-deep slots to reduce relocation efforts;
3) In tier-captive aisles, assign SKUs to tiers in ways that balance the tier/shuttle workloads;
4) In tier-to-tier aisles, assign SKUs of high affinity in the same tier to reduce tier-transfer efforts.

ABC three-class-based (3-CBS) storage assignment is a comprised policy that classifies SKU types into three classes A, B and C according to their popularity, where class A refers to the most popular ones and class C refers to the least popular ones. According to Kofler (2014), a small number of top-selling products (for instance 20%) are responsible for most picks (for instance 80%) in many warehouses, and a common 3-CBS practice for allocating the classes is a 10%-20%-70% split by popularity. Then, the SKUs are assigned with simple storage policies (e.g., COL) within the corresponding classes. Ekren et al. (2015) applied simulation modeling approaches to study tier-captive SBS/RS, and rack design criteria are explored based on simulation results under their ABC storage assumption. In their model, the lifts were assumed as the system bottleneck and the class-allocation only considers the vertical aspect, thus class A items are stored to the lower tiers (closer to lifts' I/O points) and class C to the higher tiers. As discussed earlier this section, tote lifts are more likely to be the bottlenecks in tier-captive systems (Figure 5.18), while in tier-to-tier systems the shuttles are usually the bottleneck (Figure 5.20).

Three options of allocating the ABC classes for a single aisle of tier-captive SBS/RS is proposed as illustrated in Figure 5.23. Incoming storage totes are first randomly assigned to tiers corresponding to the SKUs' classes, and then assigned according to COL within the selected tiers.

In option 1, the classes are distributed vertically, and the fast-moving products are stored to lower tiers. This option is appropriate when the tote lifts are the dominating bottlenecks of the aisle – in this way, the workloads of the tote lifts are minimized. However, the risk is that the shuttles on the lower tiers might be overloaded and become the new bottlenecks. In option 2, the classes are distributed horizontally and in the same way for all tiers – thus the shuttle workloads are balanced. This option applies to cases where the shuttles are the dominating bottlenecks of the aisle. Moreover, a third option that considers both horizontal and vertical aspects are proposed. In option 3, the classes are distributed in a radiation pattern – this is similar to common class-based storage practices in crane-based AS/RS. Option 3 is proposed for cases where the tote lift utilizations and shuttle utilizations are expected to be close at the maximum demands (thus all the devices are fully utilized, which is ideal from the design perspective discussed in Chapter 4). Essentially, option 3 seeks for a balance point between options 1 and 2. Through simulation experiments, we found that option 3 can potentially provide better system performances comparing to options 1 and 2 for Tier-captive systems. However, further study on exact methods (e.g., deciding the sizes and shapes of the classes' storage areas) still need to be explored and formalized.

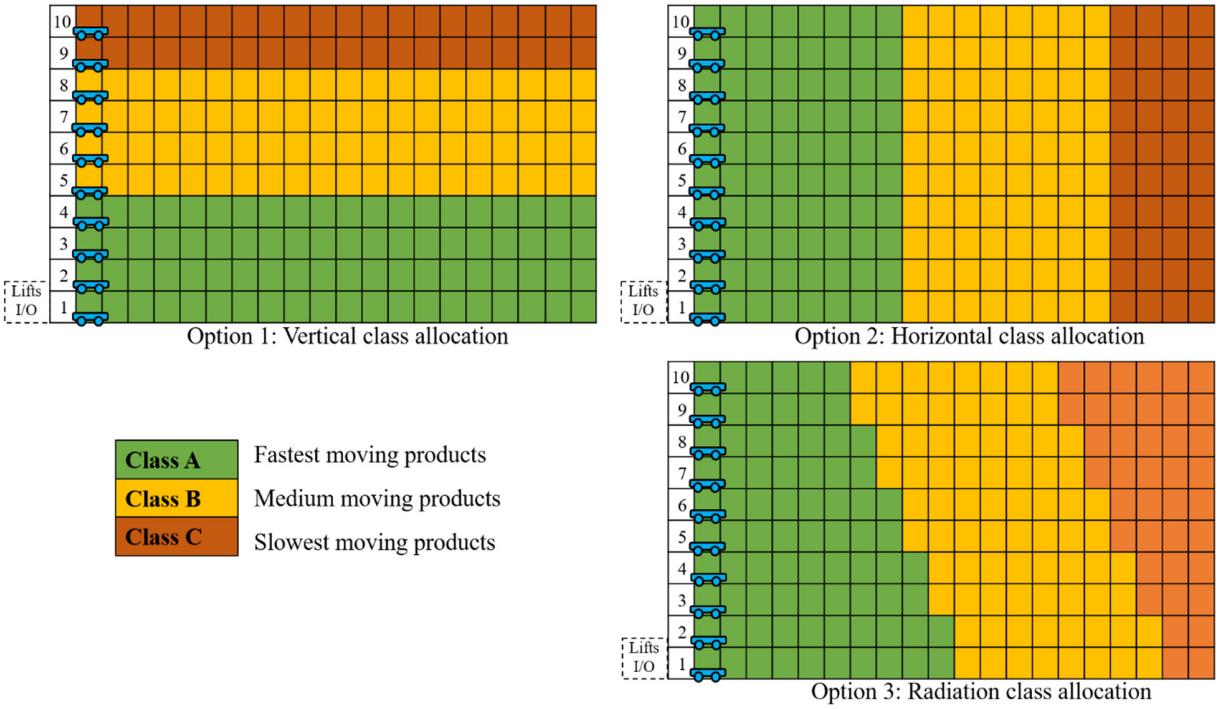

Figure 5.23 ABC Class-based Storage Options for Tier-captive SBS/RS

In a tier-to-tier aisle, J shuttles are transferred between X tiers. As discussed in Section 5.5.1, the shuttles are usually the bottlenecks of tier-to-tier aisles, thus the storage assignment approaches should aim at balancing the shuttle workloads. As discussed in the previous chapters and sections, the shuttle performances here can be improved with two goals: 1) reduce shuttle travel times on each tier; and 2) reduce occurrences of tier-transfers. With explicit SKU-level characteristics, the first goal can be achieved by applying the same horizontal class-based storage policy as the option 2 introduced above, and goal 2 can be achieved by assigning SKUs of high affinity in the same tier to reduce tier-transfer efforts.

5.5.3 Dynamic Approaches for Improving both Short-term and Long-term Performances

The COL approaches and the class-based approaches introduced previously are static storage assignment rules that mainly focus on the system's long-term performance metrics (e.g., sustainable throughput, average cycle time). During system operations, there are also control opportunities to improve the short-term performance (e.g. makespan of a set of tasks) by making use of dynamic system information and cooperating with the dynamic scheduling approaches. Based on the previous discussions, storage assignment/relocation policies based on dynamic dispatching approaches are proposed in the following section.

5.6 Control Strategy Integration Based on Dynamic Dispatching Approaches

In this section, a set of control policies based on dynamic dispatching (DD) approaches are proposed according to the observations and conclusions obtained in the previous sections. These policies involve both device scheduling and storage assignment/relocation aspects. An integrated control strategy is developed by systematically integrating different DD algorithms, as illustrated in Figure 5.24.

According to Section 5.5, Class-based COL storage policy is selected for storage assignment and relocation decisions: the SKUs and rack slots are assigned with ABC classes based on SKU popularities – the class assignment is assumed static within short periods – and COL policy is applied during operations in selecting the slots corresponding to the SKUs' classes. It is noticeable that the ABC classification is not mandatory: if SKU-level characteristics are not

explicit to the decision maker, s/he may only consider the COL aspect in storage assignment / relocation decisions. Finally, the storage lift loads as many totes as possible to its current tour from its queue, and delivers the totes to tiers in FCFS patterns.

In tier-captive systems, shuttle schedules are determined by A* Search Algorithm that the next task is selected by service time and estimated remaining makespan. This algorithm is viewed near-optimal in this research as discussed in Section 5.4.1. For tier-to-tier systems, a $P||C_{max}$-A* Algorithm is applied for tier-transfer decision-making to minimize the overall makespan of aisle tasks. As introduced in Section 5.4.2, the $P||C_{max}$-A* Algorithm takes in-tier estimates from the A* Search Algorithm as inputs, and then allocates shuttles to tiers so that to balance tier workloads with $P||C_{max}$ techniques.

As the downstream processes of the shuttle services, the retrieval lift services are applied using either Closest-First or FCFS – our previous analysis in Section 5.4 did not observe significant performance differences between these two policies.

The control strategy dynamically updates device schedules and makespan estimates whenever the system states need to be reevaluated, e.g., when a new task arrives to the aisle or a tier-transfer service is completed, or when an unexpected device failure occurs, and only updates the partial task information that is related to the state changes.

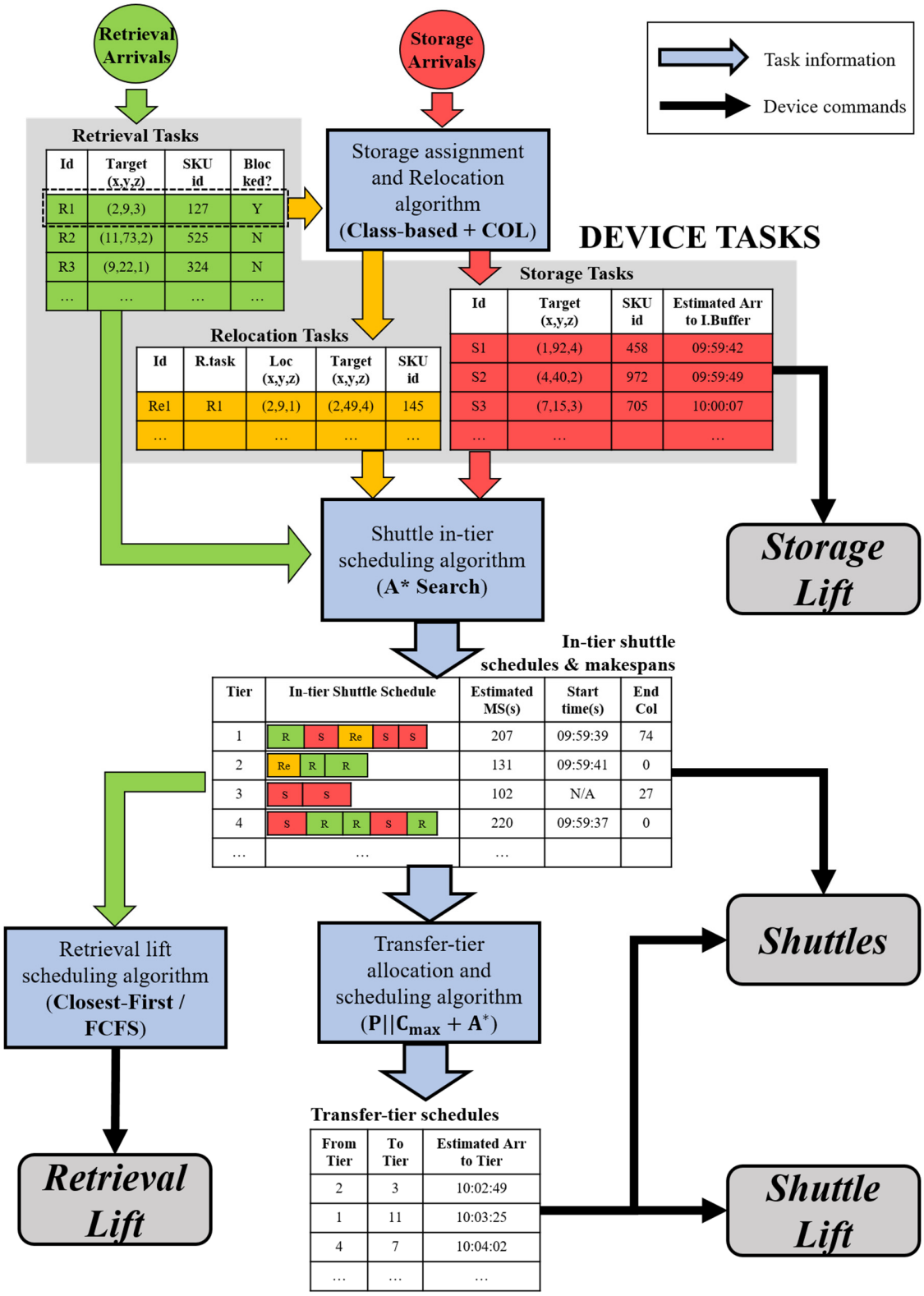

Figure 5.24 Integrated Control Strategy based on Dynamic Dispatching Algorithms

5.6.1 Dynamic Storage Assignment and Relocation

The dynamic storage assignment and relocation algorithm proposed here determines the target slot of each incoming storage tote along with the target slot of each blocker tote of blocked 2-deep retrieval tasks. In each storage task or relocation task, only one available slot is selected, and the selection is based on the tote's SKU information and the system's current state. The algorithm applies COL rule with prioritization for 2-deep slots, while also considers the device and inventory states.

The SKU-level characteristics are either assumed explicit or implicit. For the explicit case, SKUs are preassigned to one of three classes A, B, C according to turn-over rates when they arrived at the system, and each slot $[x, y, z]$ in the rack belongs to a single class. The search scope of each storage/relocation is limited to the slots in the corresponding class. In 2-deep racks, the algorithm will first check the inventory totes of the same SKU type as the target tote and attempt to assign them in neighboring deep slots so that to minimize future relocations. The logics of storage assignment and relocation are the same except for that the search space of relocation is further limited to the current tier of the blocker tote. The details of the algorithm is described in pseudocode form in Appendix III.a.

5.6.2 A* Search Algorithm for Shuttle In-tier Scheduling

The A* Search Algorithm proposed here creates proposed shuttle schedules as well as makespan estimates for in-tier storage, retrieval and relocation tasks that currently exist in the aisle, and dynamically updates those schedules and estimates whenever a new task arrives. Inputs to this algorithm include the current task set and the current states of the shuttles. Denote $f_{A*}^{V}(x, y0_x)$ as the function that creates in-tier shuttle task schedule for tier x, given the shuttle's initial location $y0_x$. Based on the existing S storage tasks, R retrieval tasks and Re relocation tasks expected to be served by the current shuttle on tier x (excluding the shuttle's current task if it is busy), schedule for in-tier shuttle tasks on each tier is developed, and the makespan is estimated. The details of the algorithm is described in pseudocode form in Appendix III.b.

Tier-transfer decisions are not made here, but the outputs of this algorithm will be used as inputs to the Tier-transfer Allocation and Scheduling algorithm (to be introduced in Section 5.6.3).

5.6.3 P||C_{max}-A* Algorithm for Tier-transfer Allocation and Scheduling

The P||C_{max}-A* Algorithm proposed here only applies to tier-to-tier systems. It creates tier-transfer plans by dynamically allocating shuttles between tiers and scheduling shuttle lift services. The algorithm make use of makespan estimates from the previous shuttle scheduling algorithm as one part of its inputs. The algorithm works in different ways depending on the shuttle fill rates in the aisle: if $J \leq X/2$, the algorithm first allocate transfer-needed tiers to the shuttles so that the shuttles' estimated workloads are balanced, and then determines the shuttle lift's pickup sequence; otherwise, the algorithm always selects the tier with most estimated workload, and then selects the shuttle with the shorted estimated travel time for the next service.

Consider a tier-to-tier system of X tiers, J shuttles ($J < X$) and T transfer-needed tiers ($T \leq X - J$). Record the following system states: the tiers that need tier-transfer services $\boldsymbol{X^T}$ and tiers currently have shuttles $\boldsymbol{X^I}$, shuttles' expected available times $t0_x^V$ and initial locations when become available $y0_x^V$ on their current tiers $x \in \boldsymbol{X^I}$, and the shuttle lift's expected available time $t0^{VL}$ and initial location when become available $x0^{VL}$. Denote M_x as the estimated service time of the tasks on each tier from the time when a shuttle is available on this tier. Based on A* Search Algorithm illustrated in Section 5.6.2, there are:

$$M_x = makespan: \begin{cases} f_{A*}^V(x, y0_x) \ , & if\ x \in \boldsymbol{X^I} \\ f_{A*}^V(x, Y)\ assuming\ \forall a_i = 0, if\ x \in \boldsymbol{X^T} \end{cases}$$

Based on the in-tier estimates and devices' start locations, the algorithm returns shuttle allocation decisions and shuttle lift schedule. The details of the algorithm is described in pseudocode form in Appendix III.c.

5.7 Summary

In this chapter, two types of control decisions identified in the previous chapter – storage assignment and device scheduling – are further studied. The command and information flows of the control decisions, as well as the complexities due to various factors (device interactions, demand/SKU characteristics, etc.), are identified and studied in depth. Control policies are developed to accommodate different system designs/configurations and operation environments, and to improve both short-term and long-term system performance.

The device scheduling algorithm is developed based on Mathematical Programming (MP) approaches and Dynamic Dispatching (DD) approaches. The development process starts with simple assumptions and established MP models using Integer Programming and Ant Colony Optimization techniques, and gradually increase model complexity to approximate practical operations. The MP solutions are viewed as baselines based on which the DD-based scheduling policies are formulated, evaluated, and fine-tuned along this incremental development process. An A* Search Algorithm and a $P||C_{max}$-A* Algorithm that customized for SBS/RS scheduling control are proposed as outcomes of this process.

The development of storage assignment/relocation decisions also follows incremental process. SKU-level characteristics are first assumed to be implicit, and a Closest-Open-Location (COL) policy that accommodates to different designs/configurations (1 and 2-deep, tier-captive and tier-to-tier) are proposed. When SKU-level characteristics are explicit, a policy that applies ABC classifications and in-class COL is proposed.

The development processes are largely based on the data-driven and data-generated simulation model introduced in Chapter 3. Finally, a control strategy that systematically integrates these algorithms is proposed for practical implementation by industrial practitioners. We observe from simulation experiments that the proposed control strategy is computationally efficient, adaptive to various system design/configurations and demand scenario assumptions, and significantly improves the system performance. In addition, the time estimates from and updated by the control algorithms are expected to provide useful information for the downstream processes of the SBS/RS.

Chapter 6 Conclusions and Future Research

6.1 Conclusions

By using separate vertical and horizontal robotic S/R devices, Shuttle-based Storage and Retrieval System (SBS/RS) technology shows great potential in improving warehousing system efficiency to adapt to the variety, stochasticity, and timing requirements brought by the rising e-commerce. However, the heterogeneous S/R devices and the multi-stage service processes of SBS/RS, the technology options involving tier-captive/ tier-to-tier configurations and multi-deep rack design, etc., as well as the stochastic and changing demands in different operational environments, all bring challenges to both the design configuration aspect and operational control aspect of SBS/RS. Aiming at facilitating both the system design practices and the control strategy development of SBS/RS-based warehouses, a comprehensive methodology is proposed by exploring research approaches including analytical modeling, simulation modeling, mathematical programming, dynamic dispatching, and statistical analysis.

In this research, simulation modeling is the key part which plays significant role in both the design methodology development and control strategy development. The simulation model is generic, data-generated, and accommodates all SBS/RS designs and technology options studied in this research. System objects and service processes, demand scenarios, as well as control decisions are modeled, and verified and validated in cooperation with our industrial sponsors. Control decisions including storage assignment, retrieval lift scheduling, shuttles in-tier scheduling, relocation, and tier-transfer are identified, and multiple alternative and configurable control approaches are implemented. In addition, the simulation techniques applied are viewed as potentially expandable to similar warehousing systems of larger scopes or/and applying different AS/R technologies.

Based on the insights obtained from the simulation modeling approach, a general analytical approach is established to support the conceptual design of SBS/RS-based warehouses. A queuing

network-based travel time model that accommodates to different system design/configuration options involving tier-to-tier configurations, different rack depths, and different tote lift capacities, etc., is developed. To improve the precision of the estimates, the queuing model also accommodates different assumptions regarding the demand scenarios and the operational control policies. A precise and efficient three-stage iterative analytical approach is developed. In the first stage, the task visit probabilities of the rack slots are estimated based on the assumption for storage policies. In the second stage, queuing analysis is conducted for each device type. The SBS/RS aisle is viewed as integrated of both the characteristics from tandem queuing systems and the characteristics from multi-server queuing systems. 12 shuttle service cases are identified, and the device utilizations and task cycle times are estimated based on the approximation of the shuttles' dual-cycle scheduling patterns as well as the approximation of the tier-transfer service patterns. In the third stage, the device utilizations and task cycle times estimated by the travel time model are evaluated, and the system's throughput and the corresponding task cycle times are found through increasing the task arrival rates iteratively. Finally, the travel time model is validated by Monte-Carlo experiments based on the simulation model. Experiment results show satisfactory precision on both the throughput and task cycle time estimates comparing with simulation results. Thus, the analytical approach is thus viewed acceptable as a precise and efficient tool for conceptual design practices.

Then, based on observations from the analytical and simulation approaches, an operational control strategy is proposed. Two types of control decisions – storage assignment and device scheduling – are further studied. The command and information flows of the control decisions, as well as the complexities due to various factors (device interactions, demand/SKU characteristics, etc.), are identified and studied in depth. Control policies are developed to accommodate different system designs/configurations and operation environments, and to improve both short-term and long-term system performance. The device scheduling algorithm is developed based on both Mathematical Programming (MP) approaches and Dynamic Dispatching (DD) approaches. The development process starts with simple assumptions and established MP models using Integer Programming and Ant Colony Optimization techniques, and gradually increase model complexity to approximate practical operations. The MP solutions are viewed as baselines based on which the DD-based scheduling policies are formulated, evaluated, and fine-tuned along this incremental development process. An A* Search Algorithm and a $P||C_{max}$-A* Algorithm that customized for

SBS/RS scheduling control are proposed as outcomes of this process. The development of storage assignment/relocation decisions also follows incremental process, and a policy that applies ABC classifications and COL policy is proposed. Finally, a control strategy that systematically integrates these algorithms is proposed for practical implementation by industrial practitioners.

6.2 Future Research

6.2.1 Order dispatching for different application environments

Besides storage assignment and device scheduling, order dispatching is viewed as another major control aspect in this research. Order dispatching determines how demand orders for SKUs are decomposed into tasks, as well as the timing of releasing those tasks to different S/R devices. Depending on the application environment, additional complexities like date requirements and precedence constraints may be introduced. In this research, for the generosity of the proposed design methodology and control strategy, simple assumptions are made regarding order structure and dispatching. However, in practical systems, the order structures could be very dependent on the application environment: as illustrated in Figure 6.1, the time constraints, SKU assortment (variety), and precedence complexity can be very different. Thus, the order dispatching approaches (and even the storage assignment and scheduling approaches) may need to be customized by the demand patterns in the application environment, and failure in doing so may lead to long waiting times and even deadlocks with S/R devices. Moreover, complex pickup operations on the workstation side may introduce further challenges to all control aspects (as illustrated in Figure 6.2). These observations provide a future research direction, while only feasible when adequate order data from different industrial applications are collected.

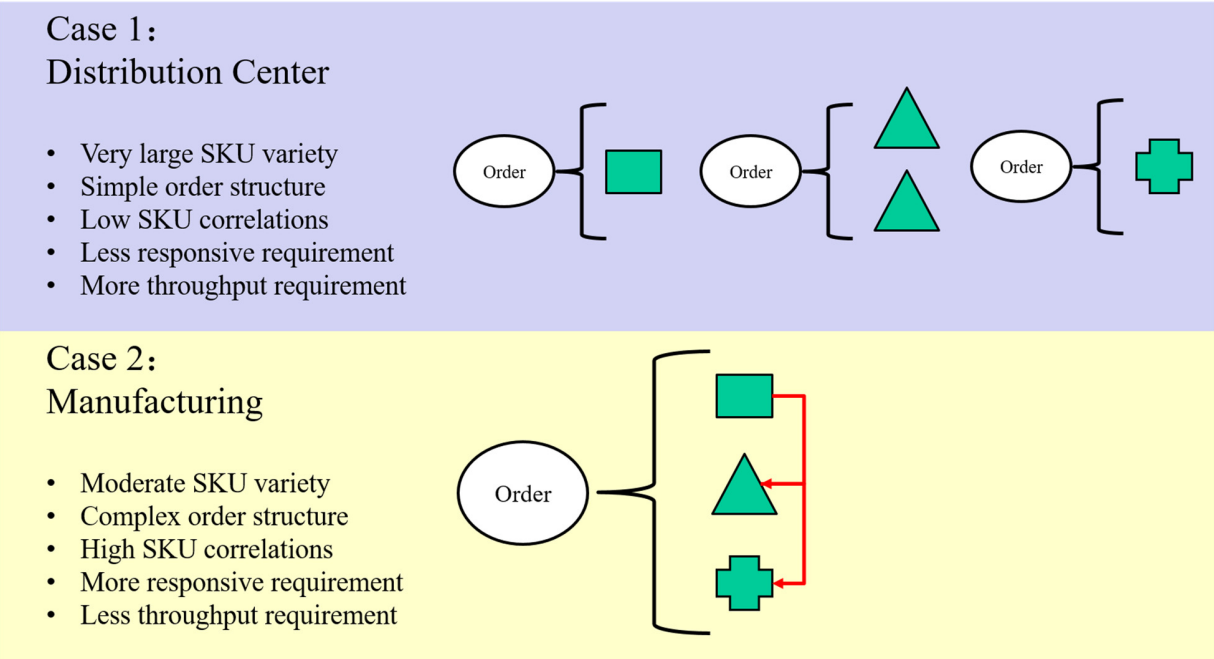

Figure 6.1 Order Structures in different application environments

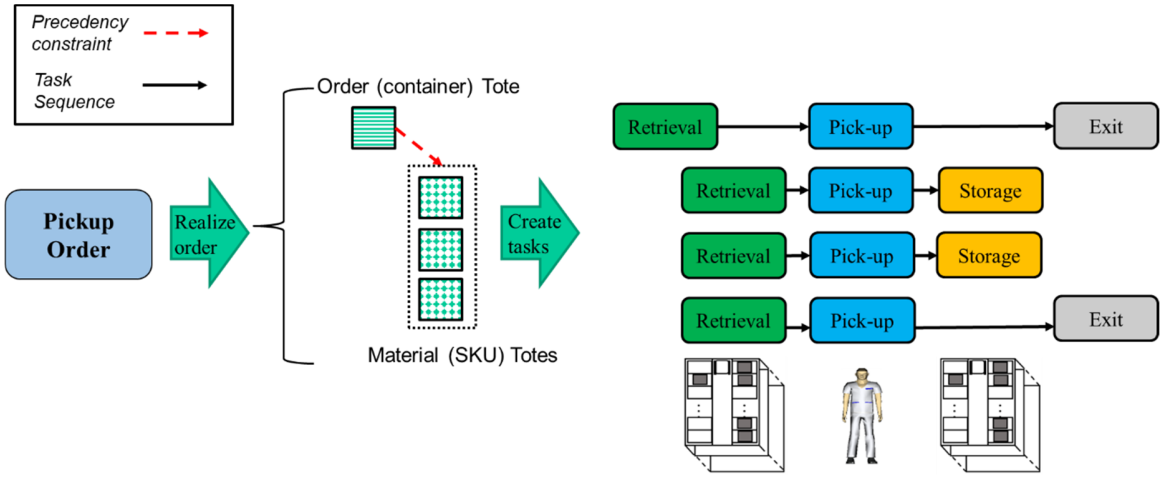

Figure 6.2 Illustration of pick-up processes in workstations

6.2.2 Global coordination with encompassing systems

In distribution center applications or manufacturing applications, the SBS/RS aisles are subsystems that interact with other subsystems upstream and downstream within the larger

distribution or manufacturing system. Depending on the features of the larger system, the operational control of the SBS/RS need to be coordinated with their encompassing subsystems to improve overall system performance and robustness. We have conducted initial research regarding the coordination of SBS/RS-based warehouses in Li et al. (2019) using simulation modelling approaches. Figure 6.3 shows a distribution center example in which pick-up processes are performed. The storage subsystem consists of multiple SBS/RS aisles, each has its own set of vehicles and lifts. Mini-load bins containing SKUs are stored in the racks of the aisles. The two-layer conveyor subsystem interacts with both the storage subsystem and the pick-up subsystem, and controls the replenishment flow of new bins. The bins are retrieved and delivered to the pick-up stations by the out-store conveyor (on 2nd floor) to fulfill customer orders. Once a pick-up order is completed, each bin may either leave the system if empty, or return to storage if not. Both the returning bins and the new bins are merged to the in-store conveyor (on 1st floor) and then stored to the aisles. In Li et al. (2019), we proposed a dynamic coordination rule to control the tote flows between the aisles and the conveyor network to avoid blocking and deadlocking (Figure 6.4).

For SBS/RS applications in manufacturing environments, the SBS/RS aisles usually play the roles of intermediate buffers between consecutive manufacturing processes. Thus, the responsiveness requirements to the SBS/RS may be more constraining than in distribution center, as delays within the racks will suspend the entire production line. Moreover, in demand scenarios where product variety is high and/or manufacturing processes are complicated, the SBS/RS control is subject to strict sorting requirements that it must not only deliver the right SKUs, but also deliver those in the correct sequence, Just-In-Time. We view the global coordination of SBS/RS with encompassing systems in different practical applications as promising research directions that could be extended in the future.

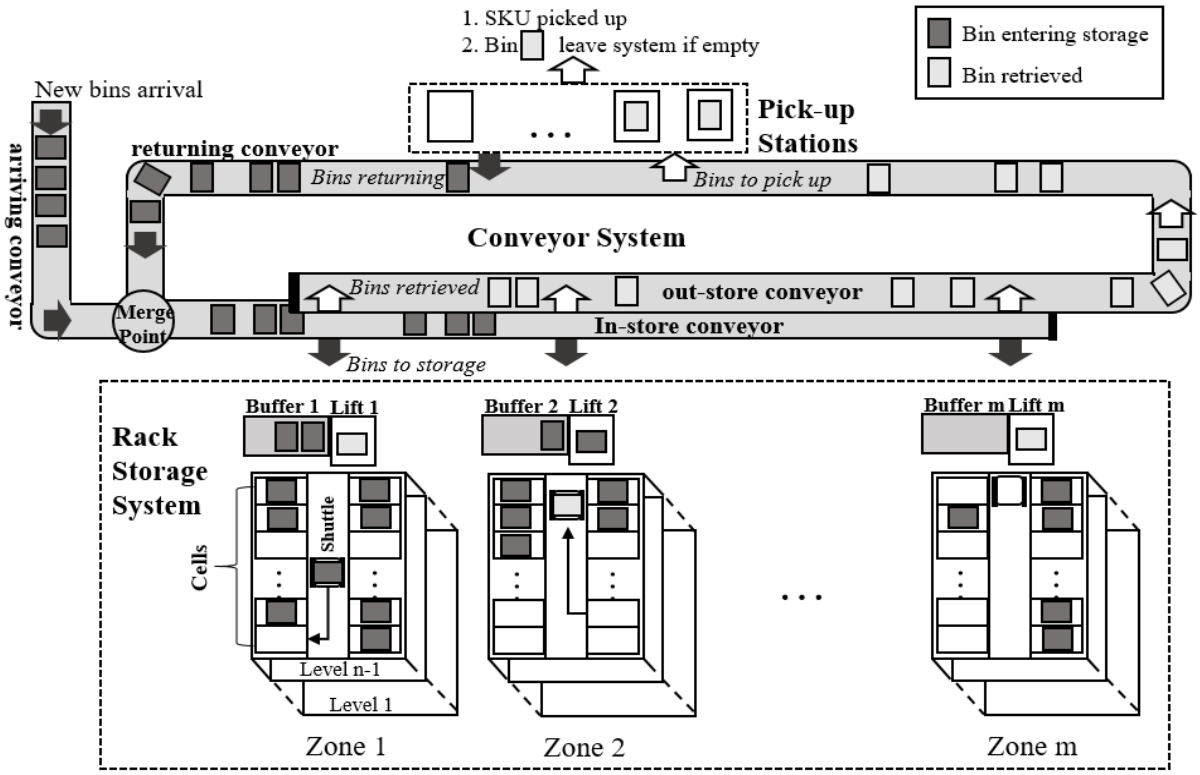

Figure 6.3 A distribution warehousing system consist of SBS/RS subsystem, Pick-up subystem, and Conveyor subsystem

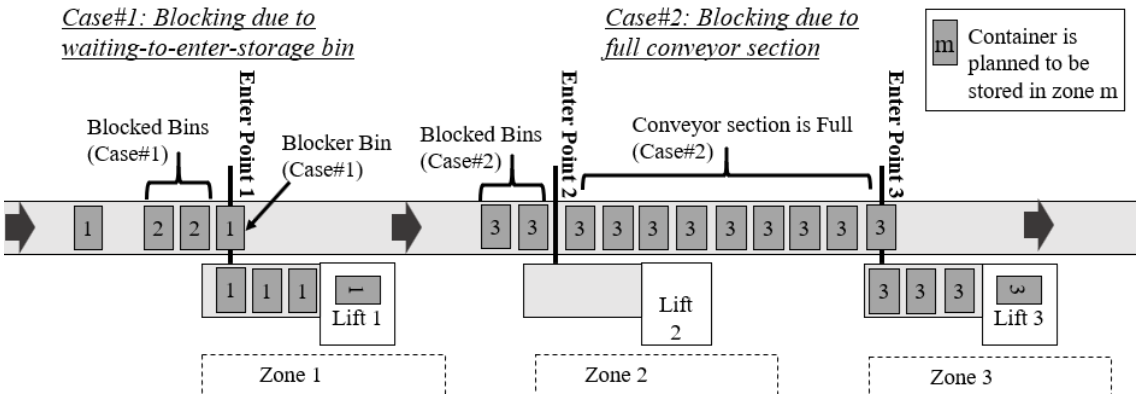

Figure 6.4 Illustration of blocking effects on conveyor network

APPENDICES

Appendix I Estimate Rack Utilization Probability Distribution Function

$I(\rho)$ Estimation based on Monte Carlo simulation Pseudocode:

$\overline{\rho}$ *# Target expected rack utilization*

T *# Time window, assume num. storages = num. retrievals within it*

$N = T\lambda$ *# Number of arrivals (each type) to be generated within T*

$K = \overline{\rho}XYZ$ *# Total slots occupied initially*

$K_x = \overline{\rho}YZ$ *# Slots occupied initially on an arbitrary tier x*

$Replications = 20$ *# An adequate number for typical designs and scenarios*

FOR ($Rep \leq Replications$)

$Generate\ N\ storage\ arrival\ times\ \boldsymbol{t^S} = \left[t_1^S, t_2^S \dots t_N^S\right]\ from\ Lognormal(\lambda, \sigma_S)$

$Generate\ N\ retrieval\ arrival\ times\ \boldsymbol{t^R} = [t_1^R, t_2^R \dots t_N^R]\ from\ Lognormal(\lambda, \sigma_R)$

$Scale\ arrival\ times: \begin{cases} t_i^S = t_i^S \times T/t_N^S \\ t_i^R = t_i^R \times T/t_N^R \end{cases}, \forall i \in [1 \dots N]$

WHILE ($\boldsymbol{t^S} \neq \emptyset$ OR $\boldsymbol{t^R} \neq \emptyset$)

IF (Next event is t_i^S)

IF ($K = XYZ$)

$Set\ t_i^S = t_{i+1}^S$

ELSE

$Remove\ t_i^S\ from\ \boldsymbol{t^S}$

$K = K + 1, record\ \rho = K/XYZ$

IF ($random(0,1) < (YZ - K_x)/(XYZ - K)$)

$K_x = K_x + 1, record\ \rho_x = K_x/YZ$

END IF

END IF

ELSE IF (Next event is t_i^R)

IF $(K = 0)$

$Set\ t_i^R = t_{i+1}^R$

ELSE

$Remove\ t_i^R\ from\ \boldsymbol{t^R}$

$K = K - 1, record\ \rho = K/XYZ$

IF $(\ random(0,1) < K_x/K\ )$

$K_x = K_x - 1, record\ \rho_x = K_x/YZ$

END IF

END IF

END IF

END WHILE

END FOR

Return probability distribution functions: $I(\rho)$ *and* $I(\rho_x)$

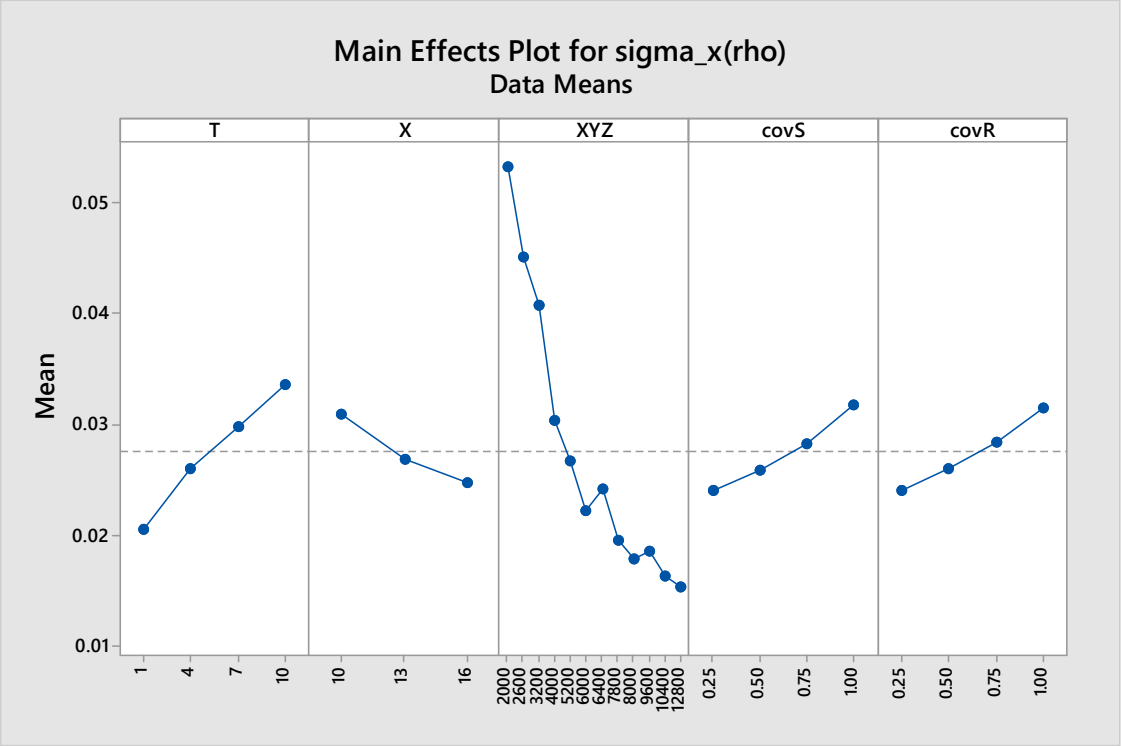

Figure A.1 Effects of design and demand parameters to $\sigma_x(\rho)$

Appendix II Scheduling and Makespan Estimation using Ant Colony Optimization

Solution Coding:

$Solution$: An array of $\forall i \in \{1 \dots R\}$ indicating a retrieval task sequence.

ACO Parameters and Variables:

K : Number of *Ants* (default value: 20). In each iteration, each ant constructs a feasible retrieval lift task sequence by randomly selecting the next task from the remaining retrieval tasks (candidates). In each selection, the probability that a task is selected is determined by the Pheromone matrix and the Visibility matrix;

P_{i_1, i_2} : *Pheromone* (trail level) for the path from task i_1 (the previous task in sequence) to i_2 (the candidate task to be added to the sequence) of the retrieval lift. A virtual task $i_1 = 0$ is defined to indicate the start of the sequence. The pheromone values updates at the end of each iteration according to the qualities of solutions found by the ants;

V_{i_1, i_2} : *Visibility* (desirability) of the path from task i_1 to i_2 in the task sequence of the retrieval lift. There is $i_1, i_2 \in \{0 \dots R\}, i_1 \neq i_2$ for all P and V.

ρ : Pheromone evaporation coefficient (default value: 0.001);

N : Maximum number of iterations without improvement in best makespan found (default value: 100000).

Pseudocode of ACO implementation

Initialize $P_{i_1, i_2} = \begin{cases} 1.0, i_1 \neq i_2 \\ 0, i_1 = i_2 \end{cases}$

Solution lower bound: $B = \max(2S\omega^{TL}, 2R\omega^{TL})$ */*baseline to evaluate solutions*/*

$Best =$ Infinity */*historical best overall makespan*/*

$BestRL =$ Infinity */*historical best retrieval lift makespan*/*

$BestVMax =$ Infinity */*historical best shuttle makespan (maximum of all shuttles)*/*

$NumNoImprovement = 0$

WHILE ($NumNoImprovement \leq N$)

FOR (Ant $k = 1,2 \dots K$)

$Candidates = \{1,2 \dots R\}$

$Solution = \emptyset$

$i_1 = 0$

$tour = 1, toursize = 0$

WHILE ($Candidates\ is\ not\ empty$)

FOR (Task i_2 in $Candidates$)

$$V_{i_1, i_2} = \begin{cases} \tau^{TL}_{0, x_{i_2}} + \omega^{TL}, & if\ tourize = 0 \\ \tau^{TL}_{x_{i_1}, x_{i_2}} + \tau^{TL}_{x_{i_2}, 0} + 3\omega^{TL}, & elsewise \end{cases}$$

IF ($x_{i_2} = x_{i_1}$)

$$V_{i_1, i_2} = V_{i_1, i_2} + \tau^{V}_{0, y_{i_2}} + \omega^{V}_{0} + \omega^{V}_{z_{i_2}}$$

END IF

$$r_{i_1, i_2} = \frac{P_{i_1, i_2} \times V_{i_1, i_2}}{\sum_{\forall i_2 \in \{candidates\}} (P_{i_1, i_2} \times V_{i_1, i_2})}$$

END FOR

$RND = random(0,1)$

$CumP = 0$

FOR (Task i_2 in $Candidates$)

IF ($RND < CumP$)

Add i_2 to $Solution$

Remove i_2 from $Candidates$

BREAK FOR

ELSE

$CumP \leftarrow CumP + r_{i_1, i_2}$

END IF

END FOR

END WHILE

FOR (Shuttle $j = 1,2 \dots J$)

Create shuttle schedules with the S_j storage tasks, R_j retrieval tasks and Re_j relocation tasks assigned to it. The schedule is developed according to **A* Search Algorithm**, and subject to constraints for

consistency with the current $Solution$, storage precedence and earliest available times, and relocation precedence;

Record makespan $F_{k,j}^{V}$ of this shuttle.

END FOR

Record maximum shuttle makespan $F_k^{VMax} = \max\left(F_{k,j}^{V}, \forall j\right)$ from this schedule.

Record completion times from shuttle services c_i^V of all retrieval tasks $i \in \{1 \dots R\}$, and assign earliest available times for retrieval lift services as $a_i^{RL} = c_i^V + t_B$;

Create retrieval lift schedule according to the sequence in the $Solution$ and subject to a_i^{RL};

Record retrieval lift makespan F_k^{RL} from this schedule;

Record ant k's solution value: $F_k = \max\left[F_k^{RL}, F_k^{VMax}\right]$

END FOR

Update pheromone matrix:

$P_{i_1, i_2} \leftarrow (1-\rho)P_{i_1, i_2} + \sum_k^K \Delta P_{i_1, i_2}^k$, where $\Delta P_{i_1, i_2}^k$ is determined as follows:

$$\Delta P_{i_1, i_2}^k = \begin{cases} \frac{10B/K}{F_k - B}, if\ F_k < Best\ or\ F_k^{RL} < BestRL\ or\ F_k^{VMax} < BestVMax \\ \frac{3B/K}{F_k - B},\ if\ F_k - Best < 0.01LB \\ \frac{B/K}{F_k - B},\ otherwise \end{cases}$$

Control pheromone values to stay above a lower limit to avoid stuck in local optimality:

$P_{i_1, i_2} = \max\left(1.0, P_{i_1, i_2}\right), if\ i_1 \neq i_2$

/ Update best RL and Shuttle makespan*/*

IF $\left(\min\left(F_k^{RL}, \forall k\right) < BestRL\right)$

$BestRL = \min\left(F_k^{RL}, \forall k\right)$

END IF

IF $\left(\min\left(F_k^{VMax}, \forall k\right) < BestVMax\right)$

$BestVMax = \min\left(F_k^{VMax}, \forall k\right)$

END IF

/ Update best overall makespan*/*

IF $\left(\min(F_k, \forall k) < Best\right)$

$Best = \min(F_k, \forall k)$

$NumNoImprovement = 0$

ELSE

$NumNoImprovement = NumNoImprovement + 1$

END IF

END WHILE

RETURN $Best$

In each iteration, each ant constructs a feasible solution randomly based on both the Visibility matrix and the Pheromone matrix. In our approach, the Visibility matrix V is evaluated dynamically according to the current tour size. As illustrated in the pseudocode, the V_{i_1, i_2} values are determined corresponding to the expected increment to the retrieval lift's makespan if candidate task i_2 is selected after the previous task i_1. If i_1, i_2 are from the same tier, V_{i_1, i_2} is further increased by the expected shuttle service time in order to prevent unnecessary waiting time on the retrieval lift's side. The solution is then mapped to the device schedules to obtain the objective function (overall makespan) in three steps. First of all, shuttle schedules are developed based on the current solution using A* Search Algorithm, subject to constraints for consistency with the retrieval sequence in the solution, storage precedence and earliest available times, and relocation precedence services (with consideration of the shuttle's earliest available time $t0_j^V$). According to the observations in the previous problems, it is assumed that each shuttle j's makespan $F_{k,j}^V$ obtained with A* Search Algorithm is near-optimal given the consistency constraints in ant k's solution. The maximum shuttle makespan F_k^{VMax} is then obtained. In the second step, the retrieval lift schedule is developed according to the sequence in the solution and subject to the task completion times from shuttle services (with consideration of the lift's earliest available time $t0^{RL}$), and the retrieval lift's makespan F_k^{RL} is obtained. In the third step, the quality of the ant's solution F_k is obtained as the maximum makespan among all devices, and the corresponding F_k^{RL} and F_k^{VMax} are also recorded.

The solutions found in each iteration are then compared to both the historical best overall makespan and the historical bests of F_k^{RL} and F_k^{VMax}, and the Pheromone matrix is updated according to the solution qualities. In our ACO implementation, a solution that is better than the historical bests in either of the makespan indicators will deposit large amount of additional

pheromone for the paths visited – we found this approach to be more effective in exploring better solutions than using a single indicator (the overall makespan). On the other hand, even when both F_k^{RL} and F_k^{VMax} are no better than the corresponding historical bests, a solution is allowed to deposit some additional pheromone if it is close to historical best in overall makespan – as such solutions may still be important for future explorations considering the rectilinear service patterns of the devices. The Pheromone of unvisited paths in the iteration are reduced by the evaporation factor. Additional lower/upper limits are set for the Pheromone matrix to prevent the heuristic from stuck in local optimal solutions. Finally, if no improvement to the historical best solution is found for the last N iterations, the ACO is terminated and the historical best solution is returned as the final solution.

The values of the ACO parameters and mechanisms here (K, N, ρ, pheromone update mechanism, etc.) are mostly determined empirically which we view as acceptably efficient and robust for various problem sizes. In determining the parameter/mechanism settings, the historical best solutions as well as Pheromone values are tracked along iterations – the settings are viewed as efficient if the best solutions are more likely to be found within fewer iterations (thus the iterations afterwards are not likely to bring further improvements). In addition, multiple ACO replications are performed to solve each problem – the settings are viewed as robust when no significant differences can be observed among the final results found by different replications (within $\pm 0.2\%$ of the replication average). As the ACO approach here is supposed to provide baselines for the development and evaluation of DD policies (instead of providing direct system control solutions), thus we will not further explore the approaches for improving the computational efficiency of the proposed ACO algorithm.

Appendix III Pseudocode of Control Strategy Algorithms

Appendix III.a Dynamic storage assignment and relocation

The algorithm applies COL rule with prioritization for 2-deep slots, while also considers the device and inventory states. Each slot is initially assigned a local priority index within its tier, illustrated as follows.

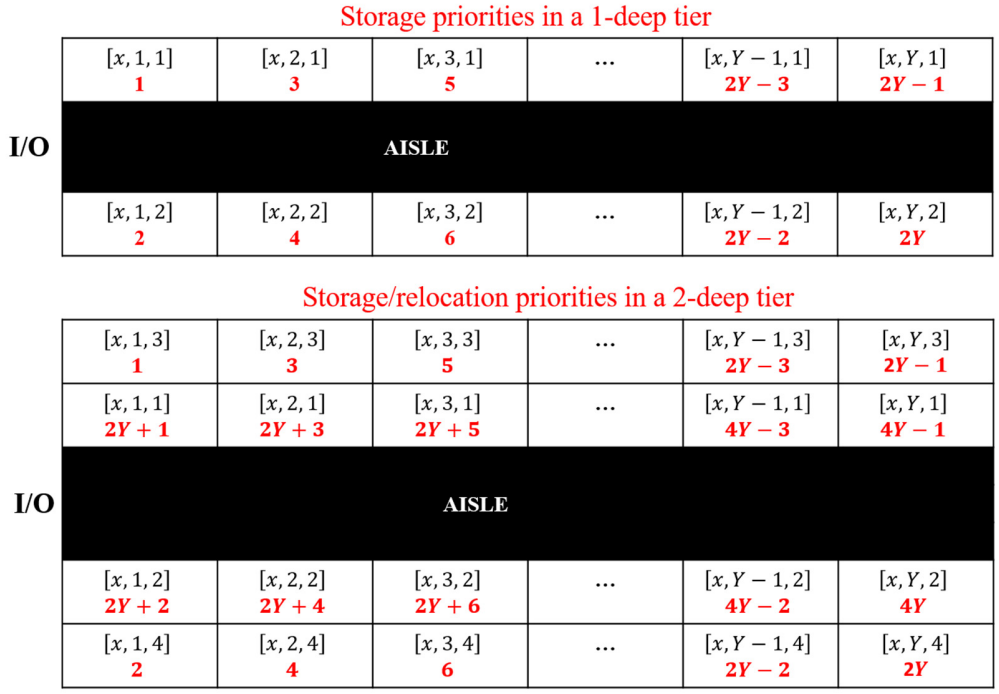

Figure A.2 Local Priority Indices for Storage assignment and Relocation

The SKU-level characteristics are either assumed explicit or implicit. For the explicit case, SKUs are preassigned to one of three classes A, B, C according to turn-over rates when they arrived at the system, and each slot $[x, y, z]$ in the rack belongs to a single class. The search scope of each storage/relocation is limited to the slots in the corresponding class. In 2-deep racks, the algorithm will first check the inventory totes of the same SKU type as the target tote and attempt to assign them in neighboring deep slots so that to minimize future relocations. The logics of storage assignment and relocation are the same except for that the search space of relocation is further limited to the current tier of the blocker tote. If SKU-level information is not explicit, the following algorithm can be applied by skipping the SKU-related evaluation logic.

Pseudocode

$SKU\ index$: i

$SKU\ class$: $k \in$ {"A", "B", "C"}

$Denote\ slots\ in\ class\ k\ as$: $\boldsymbol{A_k}$

$Denote\ tiers\ involved\ in\ \boldsymbol{A_k}\ as$: $\boldsymbol{X_k}$

$Candidates = \emptyset$

FOR (2-deep slot $[x, y, z] \in \boldsymbol{A_k}$ occupied by tote with the same SKU index i as the target tote)

IF (neighboring 1-deep slot $[x, y, z-2]$ is available)

Add neighboring 1-deep slot $[x, y, z-2]$ **To** $Candidates$

END IF

IF ($Candidates = \emptyset$)

FOR (tier x in $\boldsymbol{X_k}$)

Select best available slot $[x, y^*, z^*]$ according to COL (prioritize 2-deep)

Add $[x, y^*, z^*]$ **To** $Candidates$

END FOR

END IF

/* choose the tier of the least utilization*/

Select slot $[x^*, y^*, z^*] \in Candidates$ where $x^* = \min(\rho_x) \,|x \in \boldsymbol{X_k}$

Appendix III.b A* Search Algorithm for Shuttle In-tier Scheduling

Denote $f_{A*}^{V}(x, y0_x)$ as the function that creates in-tier shuttle task schedule for tier x, given the shuttle's initial location $y0_x$. Based on the existing S storage tasks, R retrieval tasks and Re relocation tasks expected to be served by the current shuttle on tier x (excluding the shuttle's current task if it is busy), schedules for in-tier shuttle tasks on each tier are developed, and the makespans are estimated.

The following algorithm inputs are developed:

1. Task sets:

   Storage Tasks: $\{1 \dots S\}$, indexed by arrival times to the aisle (assuming storage assignment decisions are already made). Denote task type $\pi = 1$.

   Retrieval Tasks: $\{S+1, \dots S+R\}$, in which the subset $\{S+1, \dots S+Re\}$ are tasks which need relocation. Denote task type $\pi = 2$.

   Relocation Tasks: $\{S+R+1, \dots S+R+Re\}$ with one-to-one relationship to retrieval tasks $\{S+1, \dots S+Re\}$. Denote task type $\pi = 3$.

   Denote task targets as $\{x, y_i, z_i\}$. The target of a task is the storage tote's assigned slot for each storages task, the SKU's stored slot for each retrieval task, and the blocker tote's assigned slot for each relocation task. Denote $i = 0$ as a virtual task indicating the start and end of the schedule, and assign $y_0 = y0_x$.

2. Task costs:

   d_i : The remaining service time of task i from the time the shuttle arrived its load point, computed as:

   $$d_0 = 0;$$

   $$d_i = \begin{cases} \tau^{V}_{0,\, y_i} + \omega_0^V + \omega_{z_i}^V, & if\ \pi_i = 1\ or\ 2 \\ \tau^{V}_{y_{(i-R)},\, y_i} + \omega_0^V + \omega_{z_i}^V, & if\ \pi_i = 3 \end{cases}$$

   Where $\tau^{V}_{y_1,\, y_2}$ is the shuttle travel time between columns y_1 and y_2, ω_0^V is the shuttle L/U time from/to input/output buffers, and ω_z^V is the shuttle L/U time from/to slot of depth index z.

c_{i_1, i_2} : If task i_2 is served after task i_1, the total time costs from the start (load point) of task i_1 to the start location (load point) of task i_2, including the service time of task i_1, computed as:

$$c_{i_1, 0} = d_{i_1}$$

$$c_{0,\ i_2} = \begin{cases} \tau^V_{y_0, 0} & if\ \pi_{i_2} = 1 \\ \tau^V_{y_0, y_{i_2}} & if\ \pi_{i_2} = 2 \\ \tau^V_{y_0, y_{(i_2 - R)}} & if\ \pi_{i_2} = 3 \end{cases}$$

$$c_{i_1, i_2} = \begin{cases} d_{i_1} & if\ \pi_{i_1} = 2, \pi_{i_2} = 1 \quad (R - S\ case) \\ d_{i_1} + \tau^V_{y_{i_1}, y_{i_2}} & if\ \pi_{i_1} = 1, \pi_{i_2} = 2 \quad (S - R\ case) \\ d_{i_1} + \tau^V_{y_{i_1}, 0} & if\ \pi_{i_1} = 1, \pi_{i_2} = 1 \quad (S - S\ case) \\ d_{i_1} + \tau^V_{0, y_{i_2}} & if\ \pi_{i_1} = 2, \pi_{i_2} = 2 \quad (R - R\ case) \\ \tau^V_{y_{i_1}, y_{(i_2 - R)}} & if\ \pi_{i_1} = 1, \pi_{i_2} = 3 \quad (S - Re\ case) \\ \tau^V_{0, y_{(i_2 - R)}} & if\ \pi_{i_1} = 2, \pi_{i_2} = 3 \quad (R - Re\ case) \\ \tau^V_{y_{(i_1 - R)}, 0} & if\ \pi_{i_1} = 3, \pi_{i_2} = 1 \quad (Re - S\ case) \\ \tau^V_{y_{(i_1 - R)}, y_{i_2}} & if\ \pi_{i_1} = 2, \pi_{i_2} = 2 \quad (Re - R\ case) \\ \tau^V_{y_{(i_1 - R)}, y_{(i_2 - R)}} & if\ \pi_{i_1} = 3, \pi_{i_2} = 3 \quad (Re - Re\ case) \end{cases}$$

3. Task earliest available times (relative to the time when the shuttle becomes available):

   For $\forall i \in \{1 \dots S\}$, estimate a_i based on current SL schedule and queue: if the storage tote is expected to arrive to the input buffer at time t_i (with respect to the current time), then $a_i = \max(0, t_i - t0_x)$ where $t0_x$ is the shuttle's earliest available time. Assign $a_i = 0$ for $\forall i \in \{S + 1 \dots S + R\}$ and $\forall i \in \{S + R + 1 \dots S + R + Re\}$.

Pseudocode

$Tasks = \{1 \dots S + R + Re\}$

$Scheduled = \emptyset$

$Makespan = 0$

$E_i = 0, i \in Tasks$ */* estimated task completion times */*

WHILE ($Tasks \neq \emptyset$)

$Selected = 0$

$Last = 0$

$f_{best} = INFINITY$

FOR ($i \in Tasks$)

/ Skip task in FOR loop if it cannot be started due to storage precedence or relocation-retrieval precedence */*

IF (($i \leq S$ **AND** $i > 1$) **OR** ($i \in \{S+1 \dots S+Re\}$))

SKIP i

END IF

/ Time costs g(i) to start task i after completion of the last task */*

$g(i) = \max(0,\ a_i - Time) + c_{Last,i}$

/ Estimate the remaining makespan h(i) after task i is started, by temporarily assuming Closest-First rule is applied for scheduling the remaining tasks*/*

$h(i) = 0$

$RemainingTasks = Tasks - \{i\}$

$t = Makespan + g(i)$ */* Time progress in the remaining task services */*

$i_1 = i$

WHILE ($RemainingTasks = \emptyset$)

/ Record index of the first S task i_S in remaining: all other S tasks cannot be started earlier due to storage precedence*/*

$i_S = \min(RemainingTasks)$

$Next = 0$

$f'_{best} = INFINITY$

FOR ($i_2 \in RemainingTasks$)

/ Skip task in FOR loop if it cannot be selected at the current sequence due to storage precedence or relocation precedence */*

IF (($i_2 \leq S$ **AND** $i_2 > i_S$) **OR**

($i_2 \in \{S+1 \dots S+Re\}$ **AND**

$i_2 + R \in RemainingTasks$))

SKIP i_2

END IF

/ Select the next task which is expected to cause the smallest increment to remaining makespan*/*

$f'(i_2) = \max(0,\ a_{i_2} - t)\ + c_{i_1, i_2}$

IF ($f'(i_2) < f'_{best}$)

$Next = i_2$

$f'_{best} = f'(i_2)$

END IF

END FOR

/ Update the estimates for remaining makespan*/*

Remove $Next$ **From** $RemainingTasks$

$t = t + \max(0,\ a_{Next} - t) + c_{i_1,Next}$

$i_1 = Next$

END WHILE

/ Add the service time of the last task to remaining makespan */*

$h(i) = h(i) + c_{i_1,0}$

/ Select the task with best (lowest) final compare value f = g + h */*

$f(i) = g(i) + h(i)$

IF ($f(i) < f_{best}$)

$Selected = i$

$f_{best} = f(i)$

END IF

END FOR

Add $Selected$ **To** $Scheduled$

Remove $Selected$ **from** $Tasks$

$E_i = Makespan + \max(0,\ a_{Selected} - Time) + c_{Last,Selected}$

$Makespan = E_i$

$Last = Selected$

END WHILE

/ Y: shuttle end location */*

$i_F = The\ final\ task\ in\ Scheduled$

$$Y = \begin{cases} y_{i_F}, & if\ i_F\ is\ S \\ 0, & if\ i_F\ is\ R \end{cases}$$

RETURN $Scheduled,\ Makespan,\ E_i, Y$

Appendix III.c P||C_{max}-A* Algorithm for Tier-transfer Allocation and Scheduling

Consider a tier-to-tier system of X tiers, J shuttles ($J < X$) and T transfer-needed tiers ($T \leq X - J$). Record the following system states: the tiers that need tier-transfer services $\boldsymbol{X^T}$ and tiers currently have shuttles $\boldsymbol{X^I}$, shuttles' expected available times $t0_x^V$ and initial locations when become available $y0_x^V$ on their current tiers $x \in \boldsymbol{X^I}$, and the shuttle lift's expected available time $t0^{VL}$ and initial location when become available $x0^{VL}$.

The following algorithm inputs are developed:

M_x : The estimated service time of the tasks on each tier from the time when a shuttle is available on this tier. Based on A* Search Algorithm illustrated in Section 5.6.2, there are:

$$M_x = makespan: \begin{cases} f_{A*}^V(x, y0_x) \ , & if\ x \in \boldsymbol{X^I} \\ f_{A*}^V(x, Y)\ assuming\ \forall a_i = 0, if\ x \in \boldsymbol{X^T} \end{cases}$$

Y_x : The shuttle's end location on each tier after completing this tier's task set according to A* Search Algorithm, returned from the above f_{A*}^V functions.

$D_{x_0,\, x_j, x}$: The tier-transfer time costs from the shuttle's current tier x_j to the target tier x, given the shuttle lift is initially at tier x_0. Computed as follows:

$$D_{x_0,\, x_j, x} = \tau_{Y_{x_j}, Y}^V + \tau_{x_0,\, x_j}^{VL} + \left(\tau_{x_j, x}^{VL} + 2\omega^{VL}\right)$$

Pseudocode

STEP 1: Allocate Tiers to Shuttles (only necessary for $J \leq T$ cases, otherwise skip to STEP 2)

$Allocated[j] = \emptyset, \forall j \in \{1 \dots J\}$

$ToAllocate = \boldsymbol{X^T}$

Sort $ToAllocate$ **By** M_x values in non-decreasing order.

/ estimated initial makespan of each shuttle */*

$F_j^V = t0_x^V + M_x\ , \forall x \in \boldsymbol{X^I}$

/ create an initial allocation solution by allocating transfer tiers to shuttles with minimum workloads */*

FOR ($x\ in\ \boldsymbol{X^T}$)

$j^* = j \mid \min F_j^V\ , \forall j$

$F_{j^*}^V = F_{j^*}^V + Y_x$

Add x **To** $Allocated[j]$

END FOR

$$LowerBound = \frac{\sum_{x \in \boldsymbol{X}^I}(t0_x^V + M_x) + \sum_{x \in \boldsymbol{X}^T} M_x}{J + T}$$

$StopSearch =$ FALSE

$TierCounter = 1$

$SkipShuttles = \emptyset$

WHILE ($StopSearch =$ FALSE)

/ Initialize search scope and search results */*

$Candidates = \{1 \dots J\} - SkipShuttles$

$V^{Move} = INFINITY$

$V^{Swap} = INFINITY$

Sort $Candidates$ **By** F_j^V values in non-decreasing order.

$J_{max} = Candidates[1]$

$J_{min} = Candidates[Last]$

Sort $Allocated[J_{max}]$ **By** M_x values in non-decreasing order.

/ Search for the best tier to Move from shuttle J_{max} to J_{min} in order to reduce max makespan, which is the tier with minimum makespan in J_{max} */*

IF ($TierCounter = 1$)

$x_{Move} = Allocated[J_{max}][Last]$

END IF

/ Search for the best tiers to Swap between shuttle J_{max} and J_{min} in order to reduce max makespan: x^{Swap1} denotes a single tier in J_{max} while x^{Swap2} denotes a tier set in J_{min} */*

$x_{Swap1} = Allocated[J_{max}][TierCounter]$, $W^{Swap1} = M_{x_{Swap1}}$,

$x_{Swap2} = \emptyset$, $W^{Swap2} = 0$

FOR ($x \in Allocated[J_{min}]$)

IF ($M_x + W^{Swap2} < W^{Swap1}$)

Add x **To** x_{Swap2}

$W^{Swap2} = W^{Swap2} + M_x$

ELSE

BREAK FOR

END IF

END FOR

/ Evaluate Move and Swap options by improvements to maximum makespan*/*

$V^{Move} = \max\left[\left(F^V_{J_{max}} - M_{x_{Move}}\right), \left(F^V_{J_{min}} + M_{x_{Move}}\right)\right]$

$V^{Swap} = \max\left[\left(F^V_{J_{max}} - M_{x_{swap1}} + W_{x_{swap2}}\right), \left(F^V_{J_{min}} + M_{x_{swap1}} - W_{x_{swap2}}\right)\right]$

/ Compare Move and Swap options */*

IF $(\min(V^{Move}, V^{Swap}) < F^V_{J_{max}})$

IF $(V^{Swap} < V^{Move})$

Move tier x_{move} from $Allocated[J_{max}]$ to $Allocated[J_{min}]$

$F^V_{J_{max}} = F^V_{J_{max}} - M_{x_{Move}}$

$F^V_{J_{min}} = F^V_{J_{min}} + M_{x_{Move}}$

ELSE

Swap tier x_{swap1} from $Allocated[J_{max}]$ and tier(s) x_{Swap2} from $Allocated[J_{min}]$

$F^V_{J_{max}} - M_{x_{swap1}} + W_{x_{swap2}}$

$F^V_{J_{min}} + M_{x_{swap1}} - W_{x_{swap2}}$

END IF

/ Initialize search scope for the next iteration */*

$SkipShuttles = \emptyset$

$TierCounter = 1$

ELSE

/ If no improvement is found in both options, check the shuttle with the second smallest workload */*

IF ($|SkipShuttles| < J - 1$)

Add J_{min} To $SkipShuttles$

/ If no improvement is found after all shuttles are searched, check the tier with the second largest makespan in J_{max} */*

ELSE IF ($TierCounter < |Allocated[J_{max}]|$)

$TierCounter = TierCounter + 1$

$SkipShuttles = \emptyset$

ELSE

$StopSearch =$ TRUE

END IF

END IF

END WHILE

STEP 2: Shuttle Lift Scheduling and Estimate Shuttle Makespans

$Transfers = \boldsymbol{X^T}$

$Scheduled = \emptyset$

/* location and time when each shuttle finished serving tasks on its current tier */

$x_j^V = \boldsymbol{X^I}(j),\ y_j^V = Y_{\boldsymbol{X^I}(j)},\ t_j^V = t0^V_{\boldsymbol{X^I}(j)} + M_{\boldsymbol{X^I}(j)},\ \forall j \in \{1,2 \dots J\}$

/* location and time when the shuttle lift finished its current tier-transfer service */

$x^{VL} = x0^{VL},\ t^{VL} = t0^{VL}$

$Makespan = 0$

WHILE ($Transfers \neq \emptyset$)

/* check earliest time to start the next tier-transfer service*/

$Earliest = \max\left[min\left(t_j^V\right), t^{VL}\right]$

$Denote\ x^*, j^*\ as\ the\ tier\ and\ shuttle\ selected\ for\ the\ next\ transfer\ service$

IF ($J \leq T$)

/* Apply A* Search Algorithm for $J \leq T$ cases based on allocations from STEP 1*/

$Denote\ j_x\ as\ the\ shuttle\ allocated\ to\ tier\ x\ in\ STEP\ 1$

FOR ($x \in Transfers$)

$Denote\ g(x)\ as\ the\ transfer\ completion\ time\ to\ x$

$Denote\ f(x)\ as\ the\ estimated\ total\ makespan\ if\ x\ is\ selected\ now$

$g(x) = \max\left(t_{j_x}^V - Earliest, 0\right) + D_{x^{VL}, x_{j_x}^V,\ x}$

$RemainingTiers = Transfers - \{x\}$

/* creates initial device states for remaining makespan estimation */

$rx^{VL} = x,\ rt^{VL} = g(x)$

$$\begin{cases} rx_j^V = x,\ ry_j^V = Y_x,\ rt_j^V = rt^{VL} + M_x,\ if\ j = j_x \\ rx_j^V = x_j^V,\ ry_j^V = y_j^V,\ rt_j^V = t_j^V, \quad Otherwise \end{cases}$$

/* estimate makespan by temporarily assuming the remaining tiers are selected one-at-a-time by the smallest transfer service time */

WHILE ($RemainingTiers \neq \emptyset$)

$t = \max\left[min\left(rt_j^V\right), rt^{VL}\right]$

$r^* = r \mid \min\left[\max\left(t_{j_r}^V - t, 0\right) + D_{rx^{VL}, rx_{j_r}^V,\ r}\right],$

$\forall r \in RemainingTiers$

/* update device states in the estimation */

$rt^{VL} = \max\left(t_{j_{r^*}}^V - t, 0\right) + D_{rx^{VL}, x_{j_{r^*}}^V,\ r^*}$

$rt_{j_{r^*}}^V = rt^{VL} + M_{r^*}$

$rx^{VL} = r^*,\ x_{j_{r^*}}^V = r^*,\ y_{j_{r^*}}^V = Y_{r^*}$

Remove r^* **From** $RemainingTiers$

END WHILE

/* record estimated total makespan if tier x is selected*/

$f(x) = \max\left(rt_j^V\right)$

END FOR

$x^* = x \mid \min[f(x)], \forall x \in Transfers$

$j^* = j_{x^*}$

/* record best tier x^* and shuttle j^* with smallest estimated total makespan*/

ELSE

/* Apply MaxMS-Closest policy for $J > T$ cases: select the tier with the most workload, then select the shuttle with smallest transfer service time*/

$x^* = x \mid \min[M_x], \forall x \in Transfers$

$j^* = j \mid \min\left[\max\left(t_j^V - time, 0\right) + D_{x^{VL}, x_j^V,\ x^*}\right], \forall j$

END IF

/* record shuttle lift schedule as 2-dimension array: [tier, shuttle, estimated completion] */

Add $\left[x^*, j^*, t_{j^*}^V\right]$ **To** $Scheduled$

Remove x^* **From** $Transfers$

/* update device states */

$$t^{VL} = Earliest + \max\left(t_{j^*}^{V} - time, 0\right) + D_{x^{VL}, x_{j^*}^{V},\ x^*}$$

$$t_{j^*}^{V} = t^{VL} + M_{x^*}$$

$$x^{VL} = x^*,\ x_{j^*}^{V} = x^*,\ \ y_{j^*}^{V} = Y_{x^*}$$

END WHILE

RETURN *Scheduled*

REFERENCES

Akpunar, A., Yetkin, E.B., Lerher, T., 2017. Energy efficient design of autonomous vehicle based storage and retrieval system. Journal of Applied Engineering Science 15, 25–34.

Amato, F., Basile, F., Carbone, C., Chiacchio, P., 2005. An approach to control automated warehouse systems. Control Engineering Practice 13, 1223–1241.

Andriansyah, R., Etman, L.F.P., Adan, I.J., Rooda, J.E., 2014. Design and analysis of an automated order-picking workstation. Journal of Simulation 8, 151–163.

Andriansyah, R., Etman, L.F.P., Rooda, J.E., 2010. Flow time prediction for a single-server order picking workstation using aggregate process times. International Journal on Advances in Systems and Measurements Volume 3, Number 1 & 2, 2010.

Azadeh, K., De Koster, R., Roy, D., 2019. Robotized and automated warehouse systems: Review and recent developments. Transportation Science 53, 917–945.

Bahrami, B., Piri, H. and Aghezzaf, E.H., 2019. Class-based storage location assignment: an overview of the literature. In 16th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2019) (pp. 390-397). Scitepress.

Borovinšek, M., Ekren, B.Y., Burinskienė, A., Lerher, T., 2017. Multi-objective optimisation model of shuttle-based storage and retrieval system. Transport 32, 120–137.

Boysen, N., De Koster, R., Weidinger, F., 2019. Warehousing in the e-commerce era: A survey. European Journal of Operational Research 277, 396–411.

Boysen, N., Stephan, K., 2016. A survey on single crane scheduling in automated storage/retrieval systems. European Journal of Operational Research 254, 691–704.

Bozer, Y.A., White, J.A., 1984. Travel-time models for automated storage/retrieval systems. IIE transactions 16, 329–338.

Carlo, H.J., Vis, I.F., 2012. Sequencing dynamic storage systems with multiple lifts and shuttles. International Journal of Production Economics 140, 844–853.

Chandra, C., Grabis, J., 2007, Mathematical Programming Approaches. In: Supply Chain Configuration. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-68155-9_8

Chincholkar, A.K., Chetty, O.K., 1996. Simultaneous optimisation of control factors in automated storage and retrieval systems and FMS using stochastic coloured Petri nets and the Taguchi method. The International Journal of Advanced Manufacturing Technology 12, 137–144.

Claeys, D., Adan, I., Boxma, O., 2016. Stochastic bounds for order flow times in parts-to-picker warehouses with remotely located order-picking workstations. European Journal of Operational Research 254, 895–906.

De Koster, R., Le-Duc, T., Roodbergen, K.J., 2007. Design and control of warehouse order picking: A literature review. European journal of operational research 182, 481–501.

Eben-Chaime, M., 1996. An integrative model for automatic warehousing systems. International Journal of Computer Integrated Manufacturing 9, 286–292.

Eben-Chaime, M., 1992. Operations sequencing in automated warehousing systems. The International Journal Of Production Research 30, 2401–2409.

Ekren, B., 2020. A multi-objective optimisation study for the design of an AVS/RS warehouse. International Journal of Production Research 1–20.

Ekren, B.Y., 2020. A simulation-based experimental design for SBS/RS warehouse design by considering energy related performance metrics. Simulation Modeling Practice and Theory 98, 101991.

Ekren, B.Y., 2017. Graph-based solution for performance evaluation of shuttle-based storage and retrieval system. International Journal of Production Research 55, 6516–6526.

Ekren, B.Y., Sari, Z. and Lerher, T., 2015. Warehouse design under class-based storage policy of shuttle-based storage and retrieval system. IFAC-PapersOnLine, 48(3), pp.1152-1154.

Ekren, B.Y., 2011. Performance evaluation of AVS/RS under various design scenarios: a case study.

Ekren, B.Y., Heragu, S.S., Krishnamurthy, A., Malmborg, C.J., 2014. Matrix-geometric solution for semi-open queuing network model of autonomous vehicle storage and retrieval system. Computers & Industrial Engineering 68, 78–86.

Ekren, B.Y., Heragu, S.S., Krishnamurthy, A., Malmborg, C.J., 2012. An approximate solution for semi-open queueing network model of an autonomous vehicle storage and retrieval system. IEEE Transactions on Automation Science and Engineering 10, 205–215.

Eder, M., 2020. An approach for performance evaluation of SBS/RS with shuttle vehicles serving multiple tiers of multiple-deep storage rack. The International Journal of Advanced Manufacturing Technology, 110(11), pp.3241-3256.

Fukunari, M., Malmborg, C.J., 2009. A network queuing approach for evaluation of performance measures in autonomous vehicle storage and retrieval systems. European Journal of Operational Research 193, 152–167.

Fukunari, M., Malmborg, C.J., 2008. An efficient cycle time model for autonomous vehicle storage and retrieval systems. International Journal of Production Research 46, 3167–3184.

Füßler, D., Boysen, N., 2019. High-performance order processing in picking workstations. EURO Journal on Transportation and Logistics 8, 65–90.

Gademann, A.N., 1999. Optimal routing in an automated storage/retrieval system with dedicated storage. IIE transactions 31, 407–415.

Graham, R.L., Lawler, E.L., Lenstra, J.K. and Kan, A.R., 1979. Optimization and approximation in deterministic sequencing and scheduling: a survey. In *Annals of discrete mathematics* (Vol. 5, pp. 287-326). Elsevier.

Gagliardi, J.-P., Renaud, J., Ruiz, A., 2012. Models for automated storage and retrieval systems: a literature review. International Journal of Production Research 50, 7110–7125.

Graves, S.C., Hausman, W.H., Schwarz, L.B., 1977. Storage-retrieval interleaving in automatic warehousing systems. Management science 23, 935–945.

Goetschalckx, M. and Ratliff, H.D., 1990. Shared storage policies based on the duration stay of unit loads. Management science, 36(9), pp.1120-1132.

Grigoryev, I. 2015. "AnyLogic 7 in three days". The AnyLogic Company.

Ha, Y., Chae, J., 2019. A decision model to determine the number of shuttles in a tier-to-tier SBS/RS. International Journal of Production Research 57, 963–984.

Hamzaoui, M.A., Arbaoui, T., Yalaoui, F., Sari, Z., 2021. An exact optimization method based on dominance properties for the design of AS/RSs. Transportation Research Part E: Logistics and Transportation Review 146, 102204.

Han, M.-H., McGinnis, L.F., Shieh, J.S., White, J.A., 1987. On sequencing retrievals in an automated storage/retrieval system. IIE transactions 19, 56–66.

Han, J., Pei, J. and Kamber, M., 2011. Data mining: concepts and techniques. Elsevier.

Hausman, W.H., Schwarz, L.B., Graves, S.C., 1976. Optimal storage assignment in automatic warehousing systems. Management science 22, 629–638.

Heragu, S.S., Cai, X., Krishnamurthy, A., Malmborg, C.J., 2011. Analytical models for analysis of automated warehouse material handling systems. International Journal of Production Research 49, 6833–6861.

Heragu, S.S., Cai, X., Krishnamurthy, A., Malmborg, C.J., 2009. Analysis of autonomous vehicle storage and retrieval system by open queueing network, in: 2009 IEEE International Conference on Automation Science and Engineering. IEEE, pp. 455–459.

Hu, Y.-H., Huang, S.Y., Chen, C., Hsu, W.-J., Toh, A.C., Loh, C.K., Song, T., 2005. Travel time analysis of a new automated storage and retrieval system. Computers & Operations Research 32, 1515–1544.

Hu, Y.H., Zhu, Z.D., Hsu, W.-J., 2010. Load shuffling algorithms for split-platform AS/RS. Robotics and Computer-Integrated Manufacturing 26, 677–685.

Kara, I. and Derya, T., 2015. Formulations for minimizing tour duration of the traveling salesman problem with time windows. Procedia Economics and Finance, 26, pp.1026-1034.

Kofler, M., 2014. Optimising the storage location assignment problem under dynamic conditions/eingereicht von: Monika Kofler (Doctoral dissertation, Linz).

Kuo, P.-H., Krishnamurthy, A., Malmborg, C.J., 2008. Performance modeling of autonomous vehicle storage and retrieval systems using class-based storage policies. International Journal of Computer Applications in Technology 31, 238–248.

Kuo, P.-H., Krishnamurthy, A., Malmborg, C.J., 2007. Design models for unit load storage and retrieval systems using autonomous vehicle technology and resource conserving storage and dwell point policies. Applied Mathematical Modeling 31, 2332–2346.

Lerher, T., 2016. Travel time model for double-deep shuttle-based storage and retrieval systems. International Journal of Production Research 54, 2519–2540.

Lerher, T., Ekren, B.Y., Dukic, G., Rosi, B., 2015a. Travel time model for shuttle-based storage and retrieval systems. The International Journal of Advanced Manufacturing Technology 78, 1705–1725.

Lerher, T., Ekren, Y.B., Sari, Z., Rosi, B., 2015b. Simulation analysis of shuttle based storage and retrieval systems. International journal of simulation modeling 14, 48–59.

Li, D., Smith, J.S., Li, Y., 2019. Coordinated control of multi-zone AVS/RS, conveyors and pick-up operations in warehouse system, in: 2019 Winter Simulation Conference (WSC). IEEE, pp. 2049–2060.

Linn, R., Wysk, R., 1987. An analysis of control strategies for an automated storage/retrieval system. INFOR: Information Systems and Operational Research 25, 66–83.

Malmborg, C.J., 2003. Interleaving dynamics in autonomous vehicle storage and retrieval systems. International Journal of Production Research 41, 1057–1069.

Malmborg, C.J., 2002. Conceptualizing tools for autonomous vehicle storage and retrieval systems. International journal of production research 40, 1807–1822.

Malmborg, C.J., 2001. Rule of thumb heuristics for configuring storage racks in automated storage and retrieval systems design. International Journal of Production Research 39, 511–527.

Marchet, G., Melacini, M., Perotti, S., Tappia, E., 2013. Development of a framework for the design of autonomous vehicle storage and retrieval systems. International Journal of Production Research 51, 4365–4387.

Marchet, G., Melacini, M., Perotti, S., Tappia, E., 2012. Analytical model to estimate performances of autonomous vehicle storage and retrieval systems for product totes. International Journal of Production Research 50, 7134–7148.

Monmarché, N., Guinand, F. and Siarry, P., 2010. Artificial ants (p. 576). Hoboken: Wiley-ISTE.

Ning, Z., Lei, L., Saipeng, Z., Lodewijks, G., 2016. An efficient simulation model for rack design in multi-elevator shuttle-based storage and retrieval system. Simulation Modeling Practice and Theory 67, 100–116.

Roodbergen, K.J., Vis, I.F., 2009. A survey of literature on automated storage and retrieval systems. European journal of operational research 194, 343–362.

Rosenshine, M., Chandra, M.J., 1975. Approximate solutions for some two-stage tandem queues, part 1: individual arrivals at the second stage. Operations Research 23, 1155–1166.

Roy, D., Krishnamurthy, A., Heragu, S.S., Malmborg, C.J., 2012. Performance analysis and design trade-offs in warehouses with autonomous vehicle technology. IIE Transactions 44, 1045–1060.

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

Schwarz, L.B., Graves, S.C., Hausman, W.H., 1978. Scheduling policies for automatic warehousing systems: simulation results. AIIE transactions 10, 260–270.

Tappia, E., Roy, D., De Koster, R., Melacini, M., 2017. Modeling, analysis, and design insights for shuttle-based compact storage systems. Transportation Science 51, 269–295.

Smith, J.S., 2003. Survey on the use of simulation for manufacturing system design and operation. Journal of manufacturing systems, 22(2), pp.157-171.

Tappia, E., Roy, D., Melacini, M., De Koster, R., 2019. Integrated storage-order picking systems: Technology, performance models, and design insights. European Journal of Operational Research 274, 947–965.

The SCIP Optimization Suite 8.0, Ksenia Bestuzheva, Mathieu Besançon, Wei-Kun Chen, Antonia Chmiela, Tim Donkiewicz, Jasper van Doornmalen, Leon Eifler, Oliver Gaul, Gerald Gamrath, Ambros Gleixner, Leona Gottwald, Christoph Graczyk, Katrin Halbig, Alexander Hoen, Christopher Hojny, Rolf van der Hulst, Thorsten Koch, Marco Lübbecke, Stephen J. Maher, Frederic Matter, Erik Mühmer, Benjamin Müller, Marc E. Pfetsch, Daniel Rehfeldt, Steffan Schlein, Franziska Schlösser, Felipe Serrano, Yuji Shinano, Boro Sofranac, Mark Turner, Stefan Vigerske, Fabian Wegscheider, Philipp Wellner, Dieter Weninger, Jakob Witzig, Available at Optimization Online and as ZIB-Report 21-41, December 2021

Van Den Berg, J.P., Gademann, A., 2000. Simulation study of an automated storage/retrieval system. International Journal of Production Research 38, 1339–1356.

van Gils, T., Ramaekers, K., Caris, A., de Koster, R.B., 2018. Designing efficient order picking systems by combining planning problems: State-of-the-art classification and review. European Journal of Operational Research 267, 1–15.

Zhang, L., Krishnamurthy, A., Malmborg, C.J., Heragu, S.S., 2009. Variance-based approximations of transaction waiting times in autonomous vehicle storage and retrieval systems. European Journal of Industrial Engineering 3, 146–169.