Evaluating dedicated and shared storage policies in robot-based compact storage and retrieval systems

Bipan Zou[a], René De Koster[b], Xianhao Xu[a]

[a]*School of Management, Huazhong University of Science and Technology, Wuhan, 430074, China*
[b]*Rotterdam School of Management, Erasmus University, P.O.Box 1738, 3000DR Rotterdam, The Netherlands*

Abstract

Robot-based compact storage and retrieval systems (RCSRS) have seen many implementations over the last few years. In such a system, the inventory items are stored in bins, organized in a grid. In each cell of the grid, a certain number of bins are stored on top of each other. Robots with transport and lifting capabilities move on the grid roof to transport bins between manual workstations and storage stacks. We estimate performance and evaluate storage policies of RCSRS, considering both dedicated and shared storage policies coupled with random and zoned storage stacks. Semi-open queueing networks (SOQNs) are built to estimate the system performance, which can handle both immediate and delayed reshuffling processes. We approximate the models by reduced SOQNs with two load-dependent service nodes and use the Matrix-Geometric Method (MGM) to solve them. Both simulations and a real case are used to validate the analytical models. Assuming a given number of stored products, our models can be used to optimize not only the length to width ratio of the system, but also the stack height, depending on the storage strategy used. For a given inventory and optimal system configuration, we demonstrate that the dedicated storage policy outperforms the shared storage policy in terms of dual command throughput time. However, from a cost perspective, with a maximum dual command throughput time as a constraint, we show that shared storage substantially outperforms dedicated storage. The annualized costs of dedicated storage are up to twice as large as those of shared storage, due to the larger number of storage positions required by dedicated storage and the relatively lower filling degree of storage stacks.

Keywords: Material handling; compact storage; robot technology; queueing networks; performance analysis

1. Introduction

A Robot-based compact storage and retrieval system (RCSRS) is an automated unit-load storage and order picking system using robotic technology (Fig.1). In this system, the inventoried items are stored in bins that are organized in a grid. In each cell of the grid, bins are stored on top of each other, forming a storage stack. Robots with transport and lifting capabilities move on the grid roof, transporting bins between storage stacks and workstations that are located at the bottom level of the system. Combining very high storage density (no aisles) with high flexibility due to the expandability of robot fleet and workstations, an RCSRS can be used as

Email addresses: D201377812@hust.edu.cn (Bipan Zou), rkoster@rsm.nl (René De Koster), xxhao@mail.hust.edu.cn (Xianhao Xu)

an alternative to traditional automated storage and retrieval systems. Many companies have recently started using such systems, e.g. ASDA and Ocado in the United Kingdom, Active Ants in the Netherlands, Catch of The Day in Australia, and so on, particularly in e-commerce retailers.

(a) Schematic view (b) Robot

(c) Storage stack (d) Workstation

Figure 1: Robot-based compact storage and retrieval system (Hatteland (2016))

The popularity of this system is due to a number of reasons (ElementLogic (2016a,b,c), Swisslog (2016a,b)):

1. Flexible, modular structure. A company can start with a small grid, which can be built within any existing warehouse, and then gradually expand it over time, without stopping production.
2. Flexible throughput capacity. By inserting more robots and workstations, throughput capacity can be expanded, even over a short horizon.
3. Relatively low costs. Compared with some other automated storage systems, the robots are small and relatively inexpensive.
4. Compact storage space. Bins are put on top of each other and robots transport bins on the grid roof, which eliminates travel aisles.
5. Short response times. Robots can move flexibly, in x- and y-directions, using congestion-free shortest paths between locations. This makes the system particularly attractive for e.g. e-commerce companies.

Performance analysis is important to evaluate various alternative scenarios and to determine those with the required throughput capacity and costs. The performance of an RCSRS may be affected by several factors,

including the number of robots, the system structure (width, length and height), the reshuffling processes, the robot dwell point policy, and so on. The combination of all these factors leads to a variety of system scenarios which are hard to evaluate by simulation in reasonable time. Therefore, accurate and efficient analytical performance estimation tools are needed for the potential adopters of an RCSRS.

A particular issue for many RCSRS adopters is the storage policy. As a system using high-level storage stacks, the storage policy applied in the RCSRS will determine the storage space required and the system throughput time. Specifically, storing one product in one storage stack (i.e., a dedicated storage policy) can eliminate the reshuffling of blocking bins while retrieving a requested bin, but more stacks will be required for products with large inventories, or for new incoming products. Allowing multiple products to share one storage stack (i.e., a shared storage policy) can save storage space, but the retrieval time will go up since the robot may need to reshuffle the blocking bins before it can reach the target bin. We therefore examine both dedicated and shared storage policies per stack and, in addition, random and zoned storage stacks within the grid. A zoned stack policy groups stacks storing products of the same turnover class, while within the zone products are assigned randomly. Moreover, two reshuffling processes will be investigated under the shared storage policy (see Fig.2)

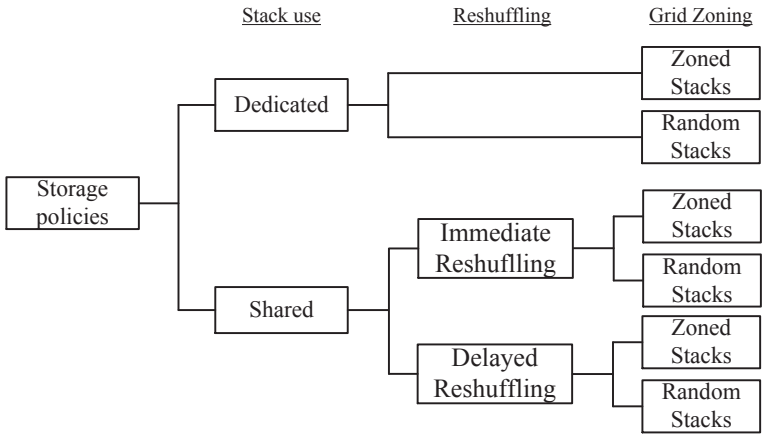


Figure 2: The storage policies and reshuffling processes investigated in this paper

This paper answers the following research questions:

1. How to build accurate and efficient analytical models for the analysis and evaluation of storage policies for an RCSRS?
2. What structure (width, length and height) is most beneficial for an RCSRS, for different storage policies?
3. When should the system use a dedicated storage policy (one stack holds one product), and when should it use a shared storage policy (multiple products share one stack)?

The system throughput time is one of the most critical performance measures of an RCSRS. It determines how long it takes to finish an order, which reflects the system service level. Also, the utilizations of robots and workstations are important performance measures. To estimate these performance measures accurately, we build SOQNs that can capture the synchronization process of robots with orders. We first approximate the original

SOQNs by reduced SOQNs with two load-dependent service nodes, using Approximate Mean Value Analysis (AMVA). Then, we use the Matrix-Geometric Method (MGM) to solve the reduced SOQNs. Analytical models are validated by both simulations and a real case. The relative errors are acceptable. For a system with a given number of stored products, we use the models to optimize the width-to-length ratio and the stack height. We also find that for a given number of products with a fixed replenishment policy, a high grid (with optimal height) fits the dedicated storage policy better, and a flat one benefits the shared storage policy more. It appears that a dedicated storage policy outperforms a shared storage policy, and the optimal zoned storage stacks outperform the random storage stacks in terms of system throughput time. However, using a cost minimization approach for the RCSRS, with a throughput time constraint, we show that shared storage can substantially reduce the system costs. The annualized costs of dedicated storage are up to twice as large as those of shared storage, due to the much larger total number of storage positions required by dedicated storage.

The remaining part of the paper is organized as follows: Section 2 presents a literature review. Section 3 describes the system and the storage and reshuffling policies. Section 4 builds the performance analysis models and Section 5 validates them by simulations and a real case. Section 6 includes the results. Section 7 draws the conclusions and suggests future work.

2. Literature review

Many papers study compact storage systems that use cranes or shuttle technology, including the 3-dimensional compact storage and retrieval systems (3DCAS), very high-density storage and retrieval systems (also called Puzzle-based storage systems PBSS), and autonomous vehicle-based storage and retrieval systems (AVS/RS) with compact storage. Different from these compact systems, an RCSRS uses robots and stores goods in bins that are organized in a grid, instead of traditional racks. We review literature on performance analysis and storage policies of these systems with an overview included in Table 1.

A 3DCAS is a compact unit-load storage and retrieval system mostly using conveyors in the depth direction and a storage and retrieval (S/R) machine at the picking face. De Koster et al. (2008) were the first to study 3DCAS. They considered a random storage policy and derived the optimal storage rack dimensions to minimize the expected travel time of the S/R machine. Yu & De Koster (2009b) investigated a two-class based storage policy and found the optimal storage zone boundaries. In addition, Yu & De Koster (2009a) studied a full-turnover storage policy and also obtained the optimal storage rack shape. Yang et al. (2015) explored the optimal storage rack dimensions, considering the acceleration and deceleration of the S/R machine. Hao et al. (2015) studied a system with I/O point at the lower mid-point of the storage rack and optimized the system dimensions. They found that the lower mid-point I/O point outperforms the lower left-corner I/O point in terms of expected travel time of the S/R machine. Zaerpour et al. (2015b) studied the storage of fresh product in a 3DCAS where the horizontal load movement is performed by satellites, instead of conveyors. They considered both dedicated and shared storage policies, and proposed a heuristic to minimize the total retrieval time with a given number of orders. The results show that the shared storage policy performs better for most practical

Table 1: Overview of literature on performance analysis and storage policies of compact storage systems

Reference	Storage area	Load movement	Storage policy
AVS/RS: Malmborg (2002); Fukunari and Malmborg (2008); Fukunari and Malmborg (2009); Heragu et al. (2011); Marchet et al. (2012); Cai, Heragu and Liu (2014); Roy et al. (2015); Lerher (2015)	multi-tier single-deep	horizontal: vehicle, vertical: lift	random
Roy et al. (2012)	single-tier, single-deep	horizontal: vehicle	random
Lerher et al. (2015)	multi-tier, double-deep	horizontal: vehicle, vertical: lift	random
Tappia et al. (2016)	single/multi-tier, multi-deep	horizontal: vehicle, vertical: lift	random
3DCAS: De Koster, Le-Duc and Yu (2008); Yang et al. (2015); Hao, Yu and Zhang (2015)	multi-tier, multi-deep	horizontal: conveyor, vertical: crane	random
Yu and De Koster (2009a)	multi-tier, multi-deep	horizontal: conveyor, vertical: crane	two-class based
Yu and De Koster (2009b)	multi-tier, multi-deep	horizontal: conveyor, vertical: crane	turnover-based
Zaerpour, Yu and De Koster (2015b)	multi-tier, multi-deep	horizontal: satellite, vertical: crane	dedicated and shared
PBSS: Gue (2006); Gue and Kim (2007); Zaerpour, Yu and De Koster (2015a)	multi-tier, multi-deep	horizontal: vehicle, vertical: lift	random
This paper:RCSRS	multi-tier stacks	horizontal and vertical: vehicle	vertical: dedicated and shared, horizontal: random and zoned

cases.

Different from a 3DCAS, a PBSS uses load-captive shuttles to transport loads horizontally, and lifts to transport loads vertically. Multiple moving shuttles can form a virtual aisle which is used to transport the requested load rapidly. Gue & Kim (2007) were the first to study a PBSS. They focused on a single-tier PBSS and derived the closed-form expression of the expected retrieval time (expressed in number of movements) for systems with one empty location located near the I/O point. Heuristics were proposed for systems with multiple empty locations located near the I/O point. Kota et al. (2015) considered a single-tier PBSS with randomly located empty locations. They derived a closed-form expression of the expected retrieval time for systems with one or two empty locations, and proposed heuristics with worst case bounds for systems with more than two empty locations. Zaerpour et al. (2015a) studied a multi-tier PBSS with many empty locations and derived closed-form formulas for the expected retrieval time for any system configuration. Moreover, they obtained the optimal system dimensions to minimize the system response time.

The AVS/RS is a new shuttle-based storage system introduced at the end of 20st century. Research on such systems uses both probabilistic and queuing models for performance estimation. Probabilistic models

can provide closed-form expressions for the single (i.e. storage or retrieval transaction) or dual-command (i.e. a combination of storage followed by retrieval) cycle time. Malmborg (2002) was the first to study AVS/R systems. He built continuous markov chain models to calculate the expected S/R machine cycle time (weighted sum of single and dual command cycle times). Lerher (2015) derived closed-form expressions for both single and dual-command cycle time. Lerher et al. (2015) extended these models to AVS/R systems with double-deep storage racks.

Due to the queuing effects between different resources in the system (vehicles, lifts), probabilistic models cannot estimate the response time of AVS/R systems accurately. Many papers have emerged using queuing models. Fukunari & Malmborg (2008) modeled the lift service process (lifts transport vehicles vertically) as an M/G/L queue (L lifts), nested within an M/G/V queue (V vehicles) that models the vehicle service process (vehicles transport loads horizontally). To address the drawbacks of nested queuing models (e.g. they cannot capture the interfaces with outside systems), Fukunari & Malmborg (2009) built closed queuing networks, while also considering maintenance and repair of vehicles.

Besides single queueing systems and closed queuing networks, open queuing networks are also used for performance estimation since they can capture the effect of waiting jobs on the system response time. Heragu et al. (2011) developed open queuing networks for AVS/R systems using tier-captive shuttles, and investigated the advantages of AVS/R systems over a traditional aisle-based automated storage and retrieval system. Marchet et al. (2012) also formulated open queuing networks to estimate the system response time of tier-captive systems, considering the acceleration and deceleration of lifts and vehicles.

Although open queuing networks can accurately estimate the system response time and throughput capacity, modeling vehicles as a circulating resource may be a better approach to investigate the effect of the number of vehicles on system performance. Based on this idea, recent studies use semi-open queuing networks for performance estimation of AVS/R systems. Roy et al. (2012) built semi-open queuing networks for a single-tier AVS/RS and developed an approximation method to derive the transaction cycle times and utilizations of vehicles and lifts. Roy et al. (2015) investigated the effect of vehicle dwell point and cross-aisle location on the system performance. They found that the end-of-aisle location of the cross-aisle is a good dwell-point location choice, as well as the load/unload point. Cai et al. (2014) studied semi-open queuing networks for multi-tier AVS/R systems and used matrix-geometric methods to derive the system performance.

The system examined in this study differs from above systems in storage area layout, load movement pattern, and storage policies. The bins in an RCSRS are organized in a grid, forming storage stacks, and are accessed by robots from the grid roof. Moreover, we consider different storage policies on the horizontal and vertical level, respectively. An RCSRS bears some resemblance with automated block stack systems (ABSS) that can be found in some warehouses, and particularly also at container terminals. Container stacks are also operated from the top, by gantry cranes. In the newest stacks, two or even three automated cranes are used per stack block with up to 5 tiers (CIMCORP (2016)). Several authors have studied storage and retrieval policies and the scheduling of cranes in ABSS (Vis & Carlo (2010), Gharehgozli et al. (2014a,b,c), Gharehgozli et al. (2015)).

However, a main difference between an ABSS and an RCSRS is that the cranes in an ABSS drive on the same rail (two directional), leading to substantial interference, while the robots in our system have more degrees of freedom (four directional), thereby reducing congestion, even with a large number of robots.

This paper makes the following contributions:

1. We are the first to study very topical robot-based storage systems consisting of block stacks operated by automated robots from the top and integrated manual workstations. We develop accurate analytical models for different storage and reshuffling policies, which can be used for system dimension optimization.
2. We investigate dedicated and shared storage policies and show that although dedicated storage outperforms shared storage in terms of dual command throughput time, from a cost perspective, shared storage is much cheaper. Therefore, in general, managers should not use dedicated storage.
3. We are the first to investigate reshuffling processes in a queueing network approach. We develop different reshuffling strategies and model the effect of reshuffling processes on system throughput time, which is shown to be substantial and cannot be neglected. Authors usually neglect reshuffling in block stacks (e.g. Gharehgozli et al. (2014b),Gharehgozli et al. (2015)).

3. System Description

Section 3.1 describes the system and presents the assumptions and notations. Section 3.2 specifies the storage policies investigated in this study. The reshuffling strategies are discussed in Section 3.3.

3.1. Robot-based compact storage and retrieval systems

An RCSRS is a compact automated storage and order-picking system, where robots (vehicles) are used to transport and store the products (Fig.1). In this system, the inventoried items are stored in standard plastic bins and a grid provides a high, dense storage space for the bins. Each cell of the grid corresponds to the entrance of a storage stack where bins are put on top of each other. Robots move on the grid roof to transport bins between storage stacks and workstations where goods enter and leave the system. The robot can lift and load bins in a storage stack, or in the workstation.

The workstations are located at the lowest level at the side of the grid (Fig.3). At the entrance of the workstation, robots can wait in a queue in a buffer area. Each workstation has one work position for the picker plus one pick-up and one drop off position for the robot. After the robot drops off a bin with items to be picked, a switching system rapidly swaps it with the previous pick bin which needs re-storage and the released bin is swapped to the picker's work position. In this way, a storage transaction immediately follows the retrieval transaction (dual-command cycles).

To facilitate the system description, we first present the main notations used in this study in Table 2.

Compared with single storage or retrieval cycles (single-command, or SC cycles), dual-command cycles (DC) can bring approximately 30% travel time reduction per cycle command in AS/R systems (Graves et al. (1977)). Also in multi-robot systems, benefits can be expected from DC cycles over SC operations. A DC cycle consists of the following steps (Fig.3):

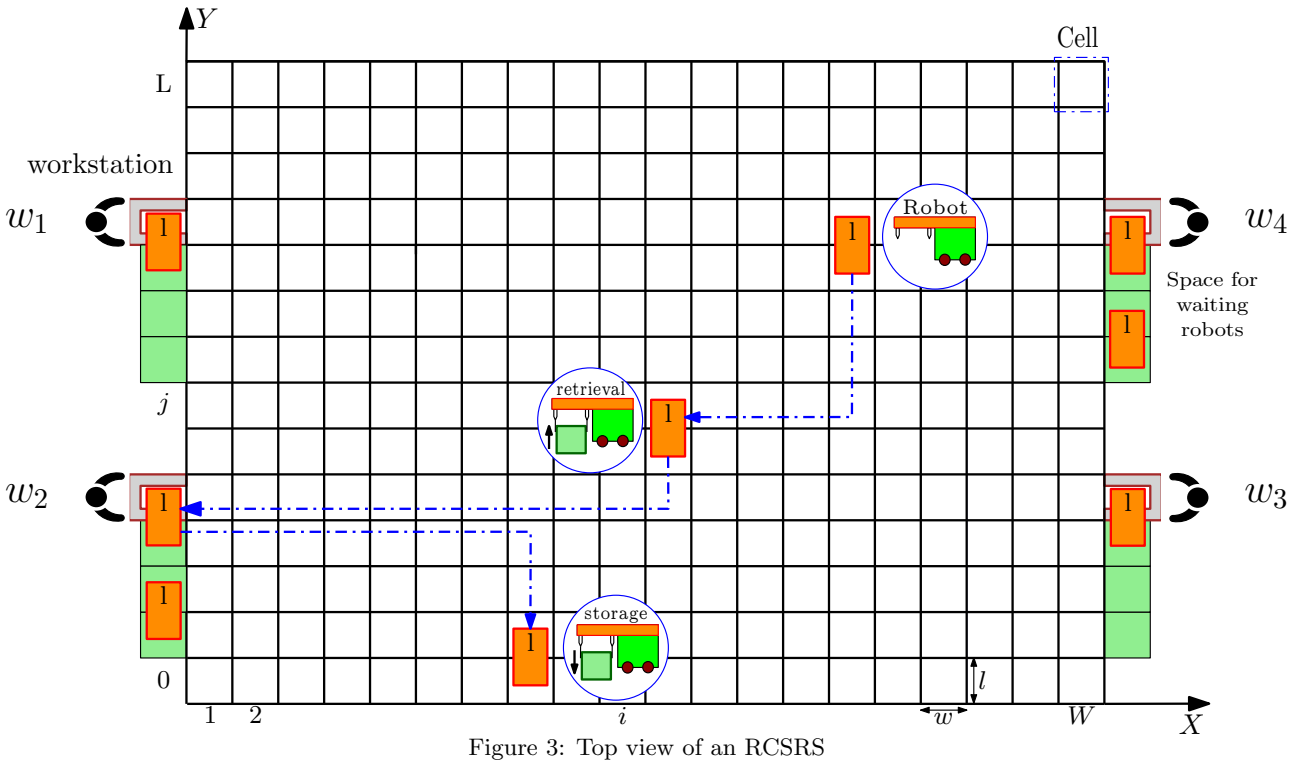


Figure 3: Top view of an RCSRS

(**a**) When a customer order arrives, it will wait in the order queue if all robots are busy, otherwise, the system will assign an available robot to it. The expected waiting time of orders for robots is W_R. We assume that a customer order can be picked from a single storage bin and the assignment of robots to orders is random. Specifically, the system will randomly select an available robot for the order, if more than one is idle. Otherwise, the first available robot will be claimed.

(**b**) The claimed robot moves from its dwell point to the retrieval position ($T_{d,r}$). We ignore the congestion and blocking effects of robots, which are minor, as the number of robots is small compared with the number of cells on the grid and the robot can choose multiple shortest paths between the starting point and the destination to avoid blocking.

(**c**) The robot digs out the retrieval bin (T_r). If the retrieval bin is at a deep level of the stack, reshuffling may be required, depending on the storage policy applied. This will be worked out in Section 3.2.

(**d**) The robot transports the retrieval bin to the designated workstation (T_{r,w_i}). We assume that the assignment rule of workstations to robots follows the random rule, i.e., any order may be picked at any workstation.

(**e**) The robot waits in the buffer area with the bin if the workstation is occupied, the expected waiting time of robots for the ith workstation is W_{w_i}. This time depends on the work time of the picker and the robot queue length.

(**f**) When the drop-off position is available, the robot drops the retrieval bin ($\frac{Hh_b}{v_l} + t_{lu}$). If the previous picking-order is finished, i.e., the picker is idle now (probability p_p^{idle}), the switching system rapidly swaps the new bin with the previous pick bin needing re-storage, and the new bin is moved to the picker's work position.

Table 2: Main Notations

Notation	Description	Notation	Description
W, L, H	System width, length (by the number of cells) and height (by the number of storage tiers)	w, l, h_b	Width and length of a cell, height of a bin (m)
v_r, v_l	Velocity of the robot and its lift (m/s^2)	t_{lu}, t_t	Time for the lift to load/unload a bin and for a robot to turn its direction (s)
C	Storage capacity, $C = LWH$	λ	Orders arrival rate (per hour)
R	Number of robots	τ	Fraction of total storage space reserved for further growth
n_w	Number of workstations. We assume the workstations are evenly distributed along the length sides, i.e., the distance of two neighboring workstations is $\frac{L}{\frac{n_w}{2}+1}$	γ	Honeycombing effect factor, $0 < \gamma < 1$. The honeycombing effect is the wasted storage space fraction due to the unavailability of a storage position to the products that are not dedicated to it.
N, N_i	Total number of products, number of products in the class that product i belongs to	N_{st}, N_{st}^i	Total number of storage stacks, number of storage stacks required by product i
D_i	Demand rate of product i, by the number of bins per unit time	K	Ratio of ordering cost to holding cost rate. Without loss of generality, we assume that K is the same for all products
SS_i	Safety stock of product i, by the number of bins	a_i	Required storage space for product i (in bins)
T_{lu}^D	Time for the robot to get the retrieval bin or release the storage bin under the dedicated storage policy	T_{lu}^S	Time for the robot to release the storage bin under the shared storage policy
T_{lu}^{SI}	Time for the robot to fetch the retrieval bin after immediate reshuffling	T_{lu}^{SD}	Time for the robot to fetch the retrieval bin after delayed reshuffling
$T_{d,r}$	Time for the robot to move from its dwell point to the retrieval position	T_r	Time for the robot to dig out the retrieval bin
T_{r,w_i}	Time for the robot to move from the retrieval position to workstation i	T_p	Time for a picker to finish an order from a bin. It is assumed to follow a uniform distribution $U[a, b]$, the same for all pickers
$T_{w_i,s}$	Time for the robot to move from workstation i to the storage position	T_s	Time for the robot to release the storage bin
w_A, w_B, w_C	Width of A, B, C zone (in cells)	P_A, P_B, P_C	Probability that the retrieval product belongs to A, B, C class
p_i	Probability that product i is to be retrieved, $p_i = \frac{D_i}{\sum_{i=1}^N D_i}$, $i = 1, 2, \cdots, N$	p_{w_i}	Probability that the order is assigned to workstation i, $i = 1, 2, \cdots, n_w$

Then, the robot loads the previous pick bin and moves to the top of the grid ($\frac{Hh_b}{v_l} + t_{lu}$). If the picker is still working on the previous picking-order, i.e., the picker is busy (probability $1 - p_p^{idle}$), the robot has to wait for the remaining pick time (denoted by T_p^{rem}) before loading the bin needing re-storage. Therefore, the service time at the workstation is $T_w = p_p^{idle}[2(\frac{Hh_b}{v_l} + t_{lu})] + (1 - p_p^{idle})[2(\frac{Hh_b}{v_l} + t_{lu}) + T_p^{rem}]$. We specify both p_p^{idle} and T_p^{rem} in next section.

(**g**) The robot transports the storage bin to the designated storage stack ($T_{w_i,s}$). Congestion and blocking effects of robots are ignored.

(**h**) The robot releases the storage bin on the top of the storage stack (T_s) and then dwells there. The robots follow the POSC (Point-of-Service-Completion) dwell point policy, i.e., they will dwell at the cell where the last order is finished.

Note that steps (**c**) and (**g**) depend on the storage policy applied, we will specify this in next section.

3.2. Dedicated vs. *shared storage policies and random* vs. *zoned storage stacks*

An RCSRS uses multi-tier storage stacks that can hold multiple bins, each for one product. We consider both dedicated (one stack holds one product) and shared (one stack holds multiple products) storage policies per storage stack. Within the grid, we consider both random (all products are randomly distributed among all stacks) and zoned (one turnover class of products is stored in the stacks within one zone) storage stacks. In the case of zoned storage stacks, we divide the products into 3 turnover classes, each of which is grouped in one zone (A, B and C) (Fig.4). The product class with the highest turnover is stored in zone A, which is located in the middle and has width w_A and N_A products. The product class with the lowest turnover is stored in zone C, which is nearby the workstations and has width w_C ($\frac{w_C}{2}$ at each side) and N_C products. The remaining products are stored in zone B, which is between zone A and zone C and has width w_B ($\frac{w_B}{2}$ at each side) and N_B products.

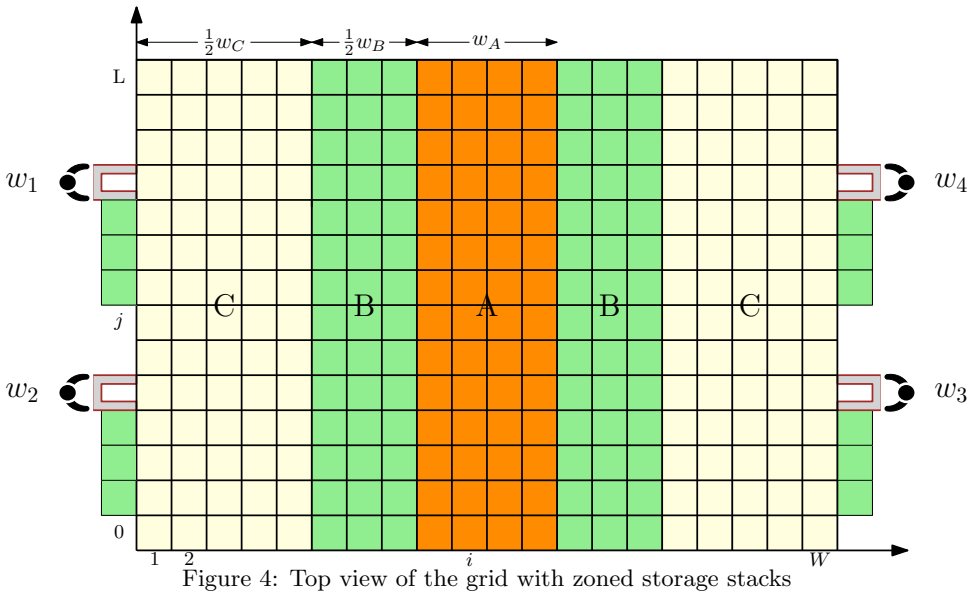


Figure 4: Top view of the grid with zoned storage stacks

Under the dedicated storage policy, one stack contains only one product, both in the cases of random and zoned storage stacks. The robot always picks up the top bin or drops off the storage bin on the top of the storage stack. So, reshuffling is avoided. However, the situation is different under the shared storage policy, and also differs between the cases of random and zoned storage stacks. In the case of random storage stacks, products are distributed randomly over all storage stacks. In the case of zoned storage stack, one turnover class of products (A, B or C) shares one zone. Reshuffling is required if the robot wants to retrieve a bin that is not on the top of the stack. In contrast, a storage bin is always dropped off on the top of a storage stack.

The storage policies applied in the storage stacks affect the robot movement on the grid roof. Let x_s be the x-axis coordinate of the stack to hold the storage bin. Under the shared storage policy coupled with zoned storage stacks, the x-axis distance between the designated workstation and the stack to hold the storage bin is less than $\frac{W}{2}$. Specifically, if the designated workstation is at the left side, $x_s = 1, 2, \cdots, \frac{W}{2}$, otherwise,

$x_s = \frac{W}{2} + 1, \frac{W}{2} + 2, \cdots, W$. Note that we assume a bin is stored in the closest zone belonging to its turnover class in the case of shared storage and zoning.

Also, the storage capacity required to store a given number of products differs between these two storage policies. Less storage space is required by the shared storage policy, due to space sharing (Yu et al. (2015)). Assuming that the Economic Ordering Quantity (EOQ) conditions hold, then, following Yu et al. (2015), the required storage space of product i, given that product i shares its storage class with N_i products in total, can be obtained by Eq.(1)

$$a_i(N_i) = 0.5(1 + N_i^{-\varepsilon})\sqrt{2KD_i} + SS_i, \tag{1}$$

where $N_i = 1$ for dedicated storage policy (for both random and zoned storage stacks), and $N_i = N$ for the shared storage policy coupled with random storage stacks. ε is the storage space sharing factor that can be obtained through Monte Carlo simulations like done by Yu et al. (2015). They found that ε is independent from the demand curves or replenishment policies, but it depends on the index of the item in the class with the largest turnover. However, it turns out that ε is rather insensitive to this index. The estimated average value of ε ranges between 0.15 and 0.25. We use the average value $\varepsilon = 0.23$, similar as in Yu et al. (2015).

Next, we will specify the number of storage stacks required by an RCSRS with height H and the average inventory level in each stack. Considering the percentage of total storage positions reserved for future growth (τ), the number of available storage positions in one stack is $\bar{H} = H(1 - \tau)$. Without loss of generality, we set $\bar{H}$ as an integer.

Under the dedicated storage policy coupled with random storage stacks, each stack can store only one product. The number of storage stacks required by product i is

$$N_{st}^i = \lceil \frac{a_i(1)}{\bar{H}} \rceil, \tag{2}$$

where $\lceil \rceil$ means rounding upward to the nearest integer.

Then, the total number of storage stacks required by the RCSRS is $N_{st} = \sum_{i=1}^{N} N_{st}^i$. Let the width to length ratio of the system be $r = \frac{W}{L}$, then, we can get the length and width of the grid by Eq.(3)

$$L = \lceil (N_{st}/r)^{\frac{1}{2}} \rceil, \; W = rL. \tag{3}$$

Under the dedicated storage policy coupled with zoned storage stacks, we have given the number of products per class N_A, N_B and N_C. Now, assume products are sorted in increasing turnover speed, the number of storage stacks required by each product class are

$$N_{st}^C = \sum_{i=1}^{N_C} N_{st}^i, \; N_{st}^B = \sum_{i=N_C+1}^{N_B+N_C} N_{st}^i, \; N_{st}^A = \sum_{i=N_B+N_C+1}^{N} N_{st}^i, \tag{4}$$

Then, the total number of storage stacks required by the RCSRS is $N_{st} = N_{st}^A + N_{st}^B + N_{st}^C$, and the length and width of the grid can also be obtained by Eq.(3). Moreover, we have $w_A = \lceil \frac{WN_{st}^A}{N_{st}} \rceil, w_B = \lceil \frac{WN_{st}^B}{N_{st}} \rceil$ and

$w_C = \lceil \frac{W N_{st}^C}{N_{st}} \rceil$.

Note that the storage positions in one stack are not always fully utilized under the dedicated storage policy (in both the random and zoned storage stack cases). The maximum number of bins stored in the stack of product i is given by

$$I_i = \lceil \frac{a_i(1)}{N_{st}^i} \rceil. \tag{5}$$

Under the shared storage policy coupled with random storage stacks, all products share the storage stacks. So, the number of storage stacks required by all products can be obtained by Eq.(6), and the length and width of the grid can be obtained by Eq.(3).

$$N_{st} = \lceil \frac{\sum_{i=1}^{N} a_i(N)}{\bar{H}} \rceil. \tag{6}$$

Under the shared storage policy coupled with zoned storage stacks, storage stacks in a specific zone are shared by the products belong to this zone. So, the number of storage stacks included in each zone can be obtained by Eq.(7), the total number of storage stacks required is $N_{st} = N_{st}^A + N_{st}^B + N_{st}^C$, and the length and width of each zone can be obtained by Eq.(3)

$$N_{st}^C = \lceil \frac{\sum_{i=1}^{N_C} a_i(N_C)}{\bar{H}} \rceil,\ N_{st}^B = \lceil \frac{\sum_{i=N_C+1}^{N_B+N_C} a_i(N_B)}{\bar{H}} \rceil,\ N_{st}^A = \lceil \frac{\sum_{i=N_B+N_C+1}^{N} a_i(N_A)}{\bar{H}} \rceil. \tag{7}$$

Since one stack can hold multiple products under the shared storage policy, the storage positions in one stack can be fully utilized. So, the number of bins stored in a storage stack (in both cases of random and zoned storage stacks) is $\bar{H}$.

3.3. Reshuffling the blocking bins

Under the shared storage policy, the robot first needs to relocate the blocking bins to other locations in order to obtain a requested deep-stored bin. We assume that the blocking bins will be put on the top of neighboring stacks, and will be returned into the same storage stack after the robot has retrieved the target bin. Two reshuffling processes are considered:

1. Immediate Reshuffling: the robot returns the blocking bins into the storage stack immediately after it has retrieved the target bin. The target bin will be put on the grid temporarily.
2. Delayed Reshuffling: the robot returns the blocking bins into the retrieval storage stack after it finishes the next storage transaction.

The blocking bins will be put on the top of neighboring stacks forming a line according to their storage sequence, i.e., the top bin at the farthest cell and the bottom bin at the cell nearby the retrieval stack. They will be returned into the original storage stack following their previous storage sequence. Without loss of generality, we assume that the blocking bins always form a line along the width direction. In the immediate

reshuffling situation, the step (**c**) of section 3.1 includes 3 components: the time to reshuffle the blocking bins on the grid roof T_{re_1}, the time to obtain the retrieval bin T_{lu}^{SI} and the time to return the blocking bins back into the storage stack T_{re_2}. In the delayed reshuffling situation, the step (**c**) contains T_{re_1} and the time to obtain the retrieval bin T_{lu}^{SD}, while another step (**i**) follows the step (**h**): the robot moves from the storage position to the retrieval position $T_{s,r}$ and then returns the blocking bins into the storage stack T_{re_2}. Note that the time for the robot to retrieve and store a bin in the storage stack may be different, depending on the storage policy applied. Under the dedicated storage policy, the time for both retrieval and storage are T_{lu}^{D}, since the robot will always access the top of the storage stack. Under the shared storage policy, the time for the robot to get the retrieval bin (after reshuffling) is T_{lu}^{SI} for immediate reshuffling policy and T_{lu}^{SD} for delayed reshuffling policy, and the time for the robot to release the storage bin is T_{lu}^{S}. We will specify these operational times in Section 4.2.

4. Performance estimation models for RCSRS

In this section, we build semi-open queueing networks to estimate the performance measures of an RCSRS. Section 4.1 develops the semi-open queueing networks (SOQN). Section 4.2 calculates the first two moments of the service time at each node of the SOQN and Section 4.3 gives the solution approaches.

4.1. Semi-open queueing network

We model the service processes of the RCSRS for dual-command orders as a SOQN. Each order consists of a storage and a retrieval transaction. The robots are modeled as an additional resource that will be matched with the arrival orders (see Fig.5). The synchronization node contains two queues where the robots are matched with orders. The idle robots wait for the orders in the robot queue (Q_R) and the arrival orders wait for the robots in the order queue (Q_O).

Table 3 gives detailed descriptions of the service nodes in the SOQN. The SOQN can handle both immediate and delayed reshuffling policies and the storage policies proposed above, by varying the structure of service nodes μ_s and μ_r. The claimed robot first moves from its dwell point to the stack that holds the retrieval bin. This is modeled as the service node $\mu_{d,r}$ with infinite servers (IS), since no waiting time is needed here. Then, the robot digs out the retrieval bin from the stack (modeled as the service node μ_r), where reshuffling of the blocking bins may be required based on the storage policy applied. Under the dedicated storage policy, one stack holds only one product; the robot always picks up the top bin and reshuffling is avoided. So, the service node μ_r only includes the retrieval operation μ_{lu}^{D}. Under the shared storage policy, to retrieve a bin from a deep level, the robot needs to reshuffle the blocking bins to the top of neighboring stacks temporarily, and then to return them into the retrieval stack by their previous storage order.

The reshuffling processes depend on the reshuffling policy applied. Under the immediate reshuffling policy, the robot first reshuffles the blocking bins on the top of neighboring stacks (service node μ_{re_1}), then retrieves the requested bin and puts it on the top of a neighboring stack (service node μ_{lu}^{SI}). Finally, it returns the blocking bins into the retrieval stack (service node μ_{re_2}). So, μ_r corresponds to three sequential nodes μ_{re_1},

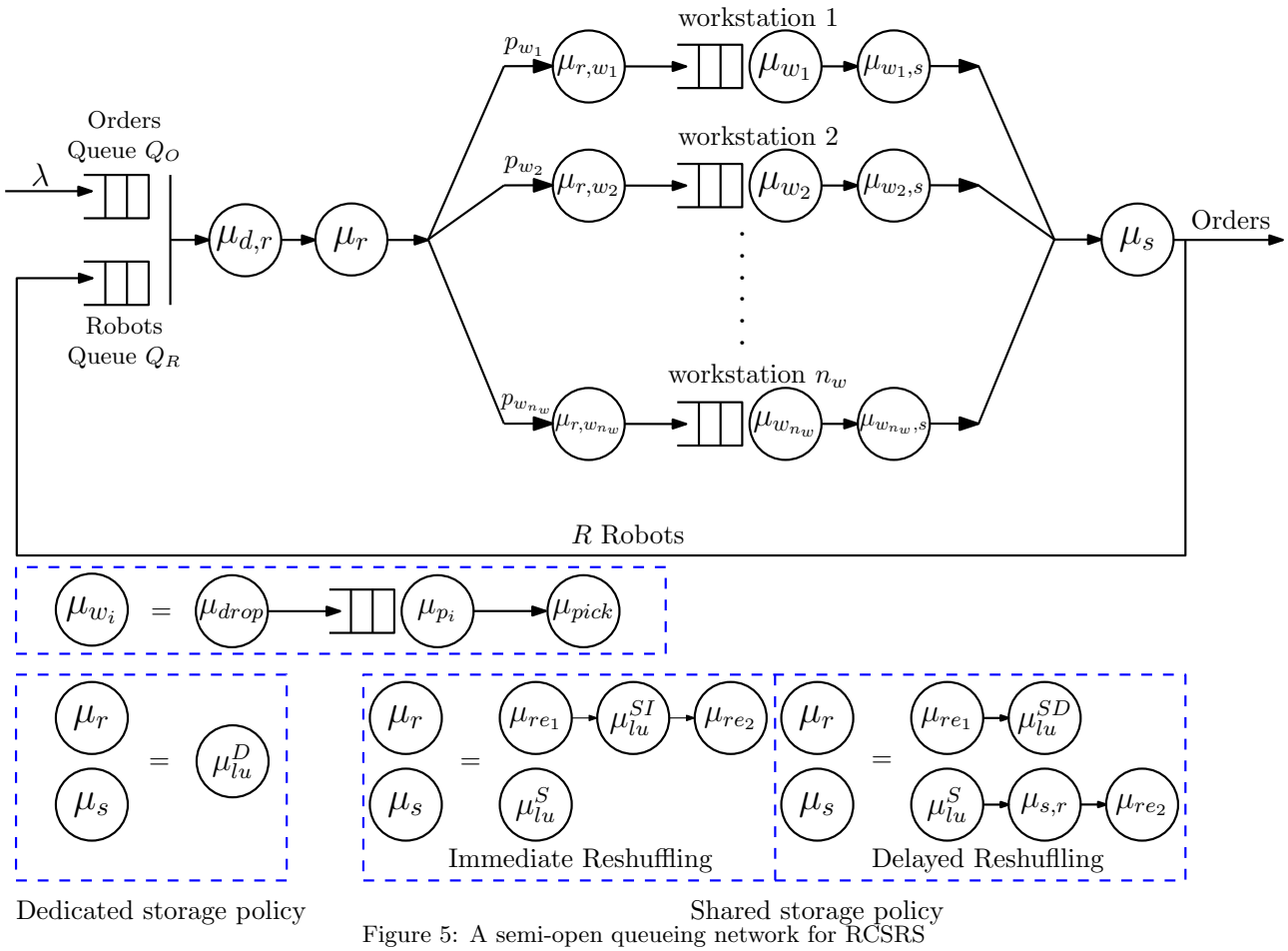


Figure 5: A semi-open queueing network for RCSRS

Table 3: Service nodes in the SOQN

Node	Description	Node	Description
$\mu_{d,r}$	Robot travels from its dwell point to the retrieval point	μ_{r,w_i}	Robot travels from the retrieval point to workstation i
μ_{w_i}	Lift drops off the retrieval bin at workstation i, then, picks up a bin needing re-storage	$\mu_{w_i,s}$	Robot travels from workstation i to the storage point
μ^D_{lu}	Lift picks up or drops off a bin from or into a storage stack under the dedicated storage policy	μ^S_{lu}	Lift drops off a bin into a storage stack under the shared storage policy
μ^{SI}_{lu}	Lift gets the retrieval bin in the case of shared storage and immediate reshuffling	μ^{SD}_{lu}	Lift gets the retrieval bin in the case of shared storage and delayed reshuffling
μ_{re_1}	Robot reshuffles the blocking bins on the top of neighboring stacks	μ_{re_2}	Robot returns the blocking bins into the retrieval storage stack
μ_{drop}	Robot drops off the retrieval bin at a workstation	μ_{pick}	Robot picks up the bin needing re-storage at a workstation
μ_{p_i}	Picker of workstation i finishes a picking order		

μ_{lu}^{SI} and μ_{re_2}. Under the delayed reshuffling policy, the robot will transport the requested bin to the designated workstation immediately upon retrieval. After finishing the storage transaction, the robot travels back to the retrieval point and returns the blocking bins into the retrieval stack. So, the service node μ_r corresponds to two sequential nodes μ_{re_1} and μ_{lu}^{SD}.

After the robot has retrieved the requested bin, it moves to the destination workstation (service node μ_{r,w_i}). Since robots are assigned randomly to workstations, the visiting probability of the service node μ_{r,w_i} is $p_{w_i=\frac{1}{n_w}}$. The robot will wait in the queue of the buffer area for the workstation to become available. Then, as specified in Section 3.1, the robot will drop off the retrieval bin and pick up the bin needing re-storage if the picker is idle, otherwise, the robot needs to wait for the remaining service time of the previous picking-order T_p^{rem}. The operational process at workstation i is modeled as the service node μ_{w_i} which consists of three components: drop off μ_{drop} (IS), picker service μ_{p_i} (single queuing system) and pick up μ_{pick} (IS). We assume that the working time of the picker on an order T_p follows a uniform distribution $U[a,b]$ and is identical for all workstations. So, $\mu_w := \mu_{w_i}$. The robot transports the storage bin to the storage position (service node $\mu_{w_i,s}$), and drops off the bin into the stack (service node μ_s). The service at node μ_s depends on the reshuffling policy applied. Specifically, it only contains the service node μ_{lu}^{S} under the immediate reshuffling policy, while it corresponds to three sequential nodes μ_{lu}^{S}, $\mu_{s,r}$ and μ_{re_2} under the delayed reshuffling policy. $\mu_{s,r}$ models the robot traveling from the storage position to the retrieval position.

The robot will be released after the completion of the order and move into Q_R while it dwells at the point of service completion. Next, we specify the calculation of the service time expressions.

4.2. Service time expressions

In this section, we calculate the service times of service nodes in the SOQN, which depend on the storage policies applied.

Let the left bottom corner of the system be the origin $(0,0)$ and the coordinates of workstation i be (x_{w_i}, y_{w_i}). Since the workstations are evenly distributed at the left and right sides of the system, the coordinates of workstations are given by Eq.(8),

$$x_{w_i} = \begin{cases} 0, & i = 1,2,\cdots,\frac{n_w}{2} \\ W, & i = \frac{n_w}{2}+1, \frac{n_w}{2}+2, \cdots, n_w \end{cases}, y_{w_i} = \begin{cases} \frac{L}{\frac{n_w}{2}+1} \cdot i, & i = 1,2,\cdots,\frac{n_w}{2} \\ \frac{L}{\frac{n_w}{2}+1} \cdot (i - \frac{n_w}{2}), & i = \frac{n_w}{2}+1, \frac{n_w}{2}+2, \cdots, n_w. \end{cases} \tag{8}$$

Denote the robot dwell point as (x_d, y_d), the coordinates of the storage stack that holds the retrieval bin and the storage stack that will hold the storage bin as (x_r, y_r) and (x_s, y_s), respectively. In the case of random storage stacks (both dedicated and shared storage policies), the coordinate distributions of the robot dwell point and the storage and retrieval positions are

$$\begin{aligned} P(x) &= \frac{1}{W}, x = 1,2,\cdots,W \\ P(y) &= \frac{1}{L}, y = 1,2,\cdots,L. \end{aligned} \tag{9}$$

In the case of zoned storage stacks (under both dedicated and shared storage policies), the coordinate distributions are different from Eq.(9). First, the probability that the retrieval item belongs to a specific class is given by

$$P_A = \frac{\sum_{i=1}^{N_A} D_i}{\sum_{i=1}^{N} D_i}, P_B = \frac{\sum_{i=N_A+1}^{N_A+N_B} D_i}{\sum_{i=1}^{N} D_i}, P_C = \frac{\sum_{i=N_A+N_B+1}^{N} D_i}{\sum_{i=1}^{N} D_i}. \tag{10}$$

The coordinate distributions of the robot dwell point and the storage and retrieval positions can now be obtained by

$$\begin{aligned} P(x) &= \begin{cases} \frac{P_C}{w_C}, & 1 \le x \le \frac{w_C}{2} \text{ or } \frac{w_C}{2} + w_B + w_A < x \le W \\ \frac{P_B}{w_B}, & \frac{w_C}{2} < x \le \frac{w_C}{2} + \frac{w_B}{2} \text{ or } \frac{w_C}{2} + \frac{w_B}{2} + w_A < x \le \frac{w_C}{2} + w_B + w_A \\ \frac{P_A}{w_A}, & \frac{w_C}{2} + \frac{w_B}{2} < x \le \frac{w_C}{2} + \frac{w_B}{2} + w_A, \end{cases} \\ P(y) &= \frac{1}{L}, 0 < y \le L. \end{aligned} \tag{11}$$

With the coordinate distributions of the robot dwell point and the storage and retrieval positions, we can calculate the service times of service stations $\mu_{d,r}$, μ_{r,w_i} and $\mu_{w_i,s}$. The travel time from the robot dwell point to the retrieval position equals

$$T_{d,r} = \frac{\mid x_d - x_r \mid \cdot w + \mid y_d - y_r \mid \cdot l}{v_r} + g \cdot t_t, \tag{12}$$

where g is the binary variable that determines whether the robot needs to turn its direction during the traveling period, and

$$g = \begin{cases} 0, & x_d = x_r \text{ or } y_d = y_r \\ 1, & x_d \neq x_r \text{ and } y_d \neq y_r. \end{cases}$$

The robot travel time from the retrieval position to workstation i, T_{r,w_i}, and the travel time from workstation i to the storage position $T_{w_i,s}$ can be calculated by Eq.(13) and Eq.(14), respectively.

$$T_{r,w_i} = \frac{\mid x_r - x_{w_i} \mid \cdot w + \mid y_r - y_{w_i} \mid \cdot l}{v_r} + g \cdot t_t. \tag{13}$$

$$T_{w_i,s} = \frac{\mid x_{w_i} - x_s \mid \cdot w + \mid y_{w_i} - y_s \mid \cdot l}{v_r} + g \cdot t_t. \tag{14}$$

With some calculation, it is now possible to obtain the first two moments of $T_{d,r}, T_{r,w_i}$ and $T_{w_i,s}$, using the probability distributions of the coordinates (see Table 10 in Appendix **A**).

As mentioned in Section 3.1, the service time at the workstation is

$$T_w = p_p^{idle}[2(\frac{Hh_b}{v_l} + t_{lu})] + (1 - p_p^{idle})[2(\frac{Hh_b}{v_l} + t_{lu}) + T_p^{rem}] = 2(\frac{Hh_b}{v_l} + t_{lu}) + (1 - p_p^{idle})T_p^{rem}.$$

To estimate the probability of the picker being idle p_p^{idle} and the remaining service time of a picking order T_p^{rem}, we approximate the service process of the picker as an $M/G/1$ queueing system with customer arrival rate

$\frac{\lambda}{n_w}$ and service time $T_p \sim U[a,b]$. The validation results in Section 5 show that the error of this approximation is acceptable. Then, we have $p_p^{idle} = 1 - \frac{\frac{\lambda}{n_w}}{\frac{2}{a+b}}$ and the mean value of T_p^{rem} is (Kleinrock (1976))

$$\bar{T}_p^{rem} = \frac{\bar{T}_p}{2}(1 + cv_p^2) = \frac{(b-a)^2 + 6(a+b)}{24}.$$

So, the first two moments of the service time at the workstation are

$$\begin{aligned} \mu_w^{-1} &= 2(\frac{Hh_b}{v_l} + t_{lu}) + (1 - p_p^{idle})\bar{T}_p^{rem}, \\ cv_w^2 &= \frac{(1 - p_p^{idle})^2 D(T_p^{rem})}{\mu_w^{-1}}, \end{aligned} \tag{15}$$

where $D(T_p^{rem})$ is the deviation of T_p^{rem}, which can be obtained by $D(T_p^{rem}) = cv_p^2 \times \bar{T}_p^{rem}$. The detailed expressions of μ_w and cv_w^2 are included in Appendix **A**.

Next, we focus on the service times of the service nodes μ_r and μ_s, which depend on the storage and reshuffling policies applied.

Under the dedicated storage policy, both service nodes μ_r and μ_s only consist of service node μ_{lu}^D. Assume that the robot needs to retrieve or store product i from or into a stack of product i with h bins ($h = 1, 2, \cdots, I_i$ where I_i is given by Eq.(5)). Then the time for the robot to retrieve or store this product i is given by

$$T_{lu}^i = \frac{2(H-h)h_b}{v_l} + t_{lu}, h = 1, 2, \cdots, I_i. \tag{16}$$

Since the probability that product i is targeted is $p_i = \frac{D_i}{\sum_{i=1}^{N} D_i}$, the first two moments of the service node μ_{lu}^D can be calculated by Eq.(17),

$$\begin{aligned} \mu_{lu^D}^{-1} &= \sum_{i=1}^{N} p_i \bar{T}_{lu}^i = \sum_{i=1}^{N} \frac{p_i}{I_i} \sum_{h=1}^{I_i} \frac{2(H-h)h_b}{v_l} + t_{lu}, \\ cv_{lu^D}^2 &= \frac{\sum_{i=1}^{N} \frac{p_i}{I_i} \sum_{h=1}^{I_i} [\frac{2(H-h)h_b}{v_l} + t_{lu}]^2 - \mu_{lu^D}^{-2}}{\mu_{lu^D}^{-1}}. \end{aligned} \tag{17}$$

Under the shared storage policy, the service processes of service nodes μ_r and μ_s consist of reshuffling operations (μ_{re_1} and μ_{re_2}) and loading/unloading operation (μ_{lu}^{SI} for the immediate reshuffling policy and μ_{lu}^{SD} for the delayed reshuffling policy). Assume that the target bin is the ith deep one (numbering from top down) of a storage stack with h bins ($i = 1, 2, \cdots, h; h = 1, 2, \cdots, \bar{H}$), then, the robot needs to reshuffle the top $i-1$ bins. The service times of service nodes μ_{re_1} and μ_{re_2} can be calculated by Eq.(18) and Eq.(19), respectively.

$$T_{re_1} = \sum_{j=1}^{i-1} [\frac{2(H-h+j)h_b}{v_l} + \frac{2jw}{v_r} + 2t_{lu}] = [\frac{2(H-h)h_b}{v_l} + 2t_{lu}](i-1) + \frac{(wv_l + h_b v_r)i(i-1)}{v_l v_r} \tag{18}$$

$$T_{re_2} = \sum_{j=1}^{i-1} [\frac{2(H-h+j+1)h_b}{v_l} + \frac{2jw}{v_r} + 2t_{lu}] = [\frac{2(H-h+1)h_b}{v_l} + 2t_{lu}](i-1) + \frac{(wv_l + h_b v_r)i(i-1)}{v_l v_r} \tag{19}$$

The first two moments of T_{re_1} and T_{re_2} can be obtained by Eq.(20) and Eq.(21), respectively.

$$\mu_{re_1}^{-1} = \bar{T}_{re_1} = \frac{1}{\bar{H}} \sum_{h=1}^{\bar{H}} \frac{1}{h} \sum_{i=1}^{h} [\frac{2(H-h)h_b}{v_l} + 2t_{lu}](i-1) + \frac{(wv_l + h_b v_r)i(i-1)}{v_l v_r}, cv_{re_1}^2 = \frac{E[T_{re_1}^2] - \bar{T}_{re_1}^2}{\bar{T}_{re_1}} \quad (20)$$

$$\mu_{re_2}^{-1} = \bar{T}_{re_2} = \frac{1}{\bar{H}} \sum_{h=1}^{\bar{H}} \frac{1}{h} \sum_{i=1}^{h} [\frac{2(H-h+1)h_b}{v_l} + 2t_{lu}](i-1) + \frac{(wv_l + h_b v_r)i(i-1)}{v_l v_r}, cv_{re_2}^2 = \frac{E[T_{re_2}^2] - \bar{T}_{re_2}^2}{\bar{T}_{re_2}} \quad (21)$$

Under the immediate reshuffling policy, the robot will just pick up the retrieval bin in the storage stack, puts it on a neighboring cell and moves back to load it after the reshuffling is finished. The operational time is

$$T_{lu}^{SI} = \frac{2(H-h+i)h_b}{v_l} + t_{lu} + \frac{l}{v_r} + t_{lu} + \frac{2l}{v_r} + t_{lu}, \quad (22)$$

Under the delayed reshuffling policy, the operations for the robot to load the retrieval bin corresponds to the robot will picks up the retrieval bin from the storage stack. The operational time is

$$T_{lu}^{SD} = \frac{2(H-h+i)h_b}{v_l} + t_{lu}. \quad (23)$$

The first two moments of T_{lu}^{SI} and T_{lu}^{SD} can now be obtained as follows

$$\mu_{lu^{SI}}^{-1} = \bar{T}_{lu}^{SI} = \frac{1}{\bar{H}} \sum_{h=1}^{\bar{H}} \frac{1}{h} \sum_{i=1}^{h} \frac{2(H-h+i)h_b}{v_l} + 3t_{lu} + \frac{3l}{v_r}, cv_{lu^{SI}}^2 = \frac{E[T_{lu}^{SI^2}] - \bar{T}_{lu}^{SI^2}}{\bar{T}_{lu}^{SI}} \quad (24)$$

$$\mu_{lu^{SD}}^{-1} = \bar{T}_{lu}^{SD} = \frac{1}{\bar{H}} \sum_{h=1}^{\bar{H}} \frac{1}{h} \sum_{i=1}^{h} \frac{2(H-h+i)h_b}{v_l} + t_{lu}, cv_{lu^{SD}}^2 = \frac{E[T_{lu}^{SD^2}] - \bar{T}_{lu}^{SD^2}}{\bar{T}_{lu}^{SD}} \quad (25)$$

The first two moments of the service time at each service node are presented in Appendix **A**, Table 10.

4.3. Solution approaches for the SOQN

The SOQN has a single customer and multiple service nodes with general distributed service times. Unfortunately, no product-form solutions are available for such SOQNs (Jia & Heragu (2009)). In this section, we therefore use the following approximate method (called *AM-SOQN*) to solve the SOQN:

1. We build a closed queueing network made up of service nodes μ_{r,w_i}, μ_w and $\mu_{w_i,s}$ (Fig.6). Then, we derive the load-dependent throughput of the closed queueing network, denoted by $TH_1(n_1)$, using the Approximate Mean Value Method (AMVA) (Appendix **B**), and replace this part by a composite service node with load-dependent service rates $\mu_{c_1}(n_1) = TH_1(n_1)$.
2. We build a closed queueing network made up of service nodes $\mu_{d,r}$, μ_r and μ_s, derive its load-dependent throughput $TH_2(n_2)$. We replace this part by a composite service node with load-dependent service rates $\mu_{c_2}(n_2) = TH_2(n_2)$ (Fig.7).
3. We now obtain a reduced semi-open queueing network with two load-dependent service nodes (Fig.8), and solve it by the Matrix-Geometric Method (MGM).

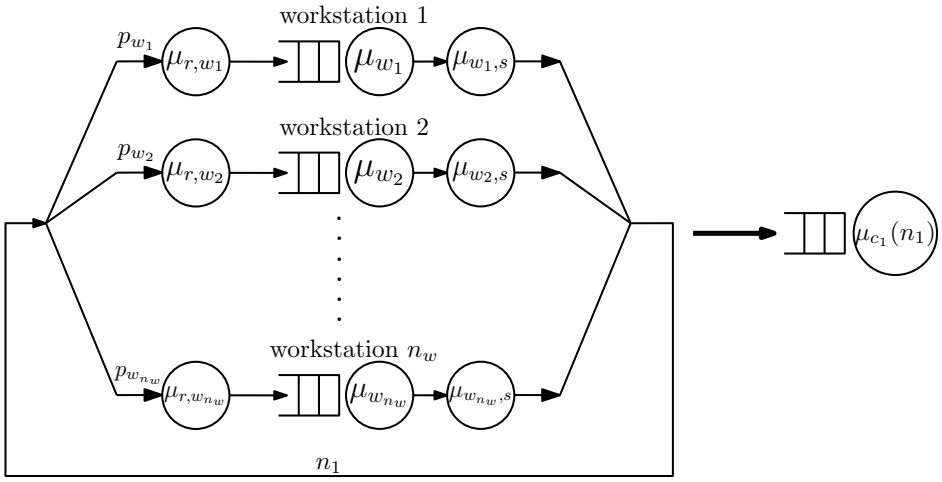


Figure 6: A closed queueing network for the first composite service node

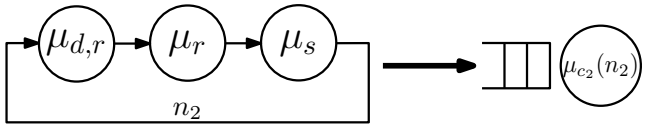


Figure 7: A closed queueing network for the second composite service node

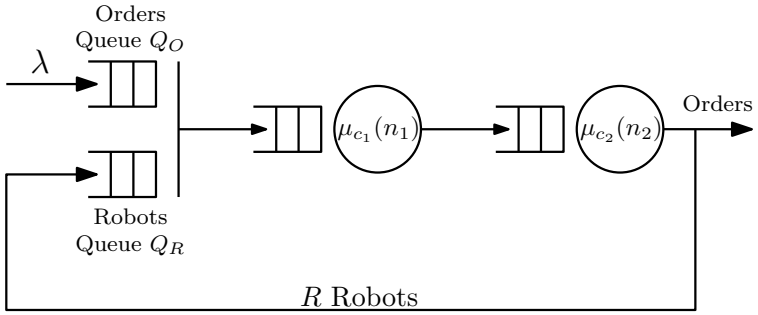


Figure 8: The reduced semi-open queueing network

We define the state variable of the reduced SOQN as $s_k = (n_{oq}, n_1, n_2)$, where n_{oq} is the number of waiting orders in the orders queue and the state index k is given by Eq.(26)

$$k = \begin{cases} \frac{(n_1+n_2)(n_1+n_2+1)}{2} + n_2, n_{oq} = 0 \\ n_{oq}(R+1) + \frac{R(R+1)}{2} + n_2, n_{oq} > 0. \end{cases} \tag{26}$$

The state transition processes is depicted in Fig.9.

Let π_k be the steady state probability of state s_k. We define the steady state probability vectors as $\boldsymbol{\pi_0} = (\pi_0, \pi_1, \cdots, \pi_{\frac{R(R+3)}{2}})$ and $\boldsymbol{\pi_i} = (\pi_{\frac{R(R+3)}{2}+(i-1)(R+1)+1}, \cdots, \pi_{\frac{R(R+3)}{2}+i(R+1)})$, $i \geq 1$, where the index of

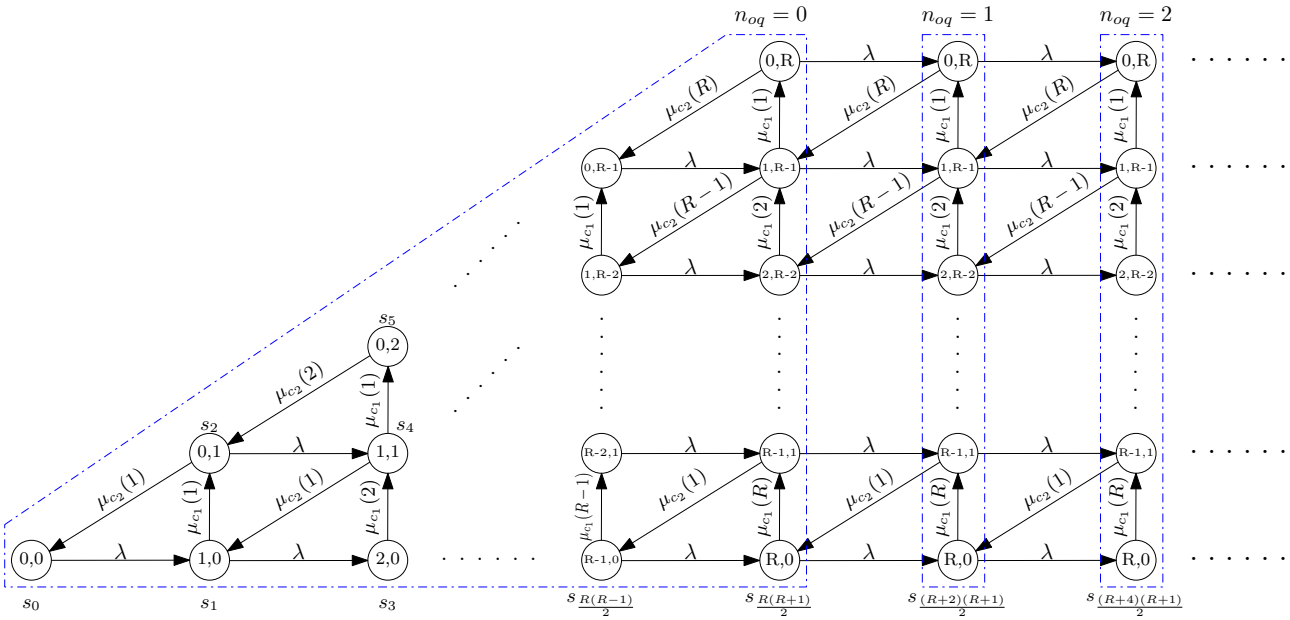


Figure 9: The state transition process

$\boldsymbol{\pi_i}$ represents that i orders are waiting in the order queue. Then, the generator matrix equals

$$\mathbf{Q}=\begin{bmatrix} \mathbf{B}_{00} & \mathbf{B}_{01} & 0 & 0 & 0 & \cdots \\ \mathbf{B}_{10} & \mathbf{B} & \mathbf{A} & 0 & 0 & \cdots \\ 0 & \mathbf{C} & \mathbf{B} & \mathbf{A} & 0 & \cdots \\ 0 & 0 & \mathbf{C} & \mathbf{B} & \mathbf{A} & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}, \tag{27}$$

where $\mathbf{B}_{00}$ is the transition matrix from $\boldsymbol{\pi_0}$ to $\boldsymbol{\pi_0}$, $\mathbf{B}_{01}$ is the transition matrix from $\boldsymbol{\pi_0}$ to $\boldsymbol{\pi_1}$, $\mathbf{B}_{10}$ is the transition matrix from $\boldsymbol{\pi_1}$ to $\boldsymbol{\pi_0}$, $\mathbf{A}$ is the transition matrix from $\boldsymbol{\pi_i}$ to $\boldsymbol{\pi_{i+1}}$, $\mathbf{B}$ is the transition matrix from $\boldsymbol{\pi_i}$ to $\boldsymbol{\pi_i}$ and $\mathbf{C}$ is the transition matrix from $\boldsymbol{\pi_{i+1}}$ to $\boldsymbol{\pi_i}$ (The details of these sub-matrixes are given in Appendix **C**).

With the generator matrix $\mathbf{Q}$, we can obtain the so-called rate matrix $\mathbf{R}$ by the iterative procedures presented in Appendix **C**, based on Eq.(28)

$$\mathbf{A}+\mathbf{R}\cdot\mathbf{B}+\mathbf{R}^{\mathbf{2}}\cdot\mathbf{C}=\mathbf{0} \tag{28}$$

From this, we can find the steady state probability vectors $\boldsymbol{\pi_0}$ and $\boldsymbol{\pi_1}$ by solving the equation set (29)

$$\begin{cases} \begin{bmatrix} \boldsymbol{\pi_0} & \boldsymbol{\pi_1} \end{bmatrix} \begin{bmatrix} \mathbf{B}_{00} & \mathbf{B}_{01} \\ \mathbf{B}_{10} & \mathbf{B}+\mathbf{R}\cdot\mathbf{C} \end{bmatrix} = \mathbf{0} \\ \begin{bmatrix} \boldsymbol{\pi_0} & \boldsymbol{\pi_1} \end{bmatrix} \begin{bmatrix} \mathbf{e}_{\frac{(R+1)(R+2)}{2}\times 1} \\ (\mathbf{I}-\mathbf{R})^{-1}\cdot\mathbf{e}_{(R+1)\times 1} \end{bmatrix} = 1, \end{cases} \tag{29}$$

where $\mathbf{e}$ is the unit vector.

The other steady state probability vectors can be obtained by $\boldsymbol{\pi_{i+1}} = \boldsymbol{\pi_i R}$, $i = 1, 2, \cdots$.

The expected number of waiting orders in the orders queue can now be obtained by Eq.(30)

$$L_{oq} = \sum_{i=1}^{+\infty} \mid \boldsymbol{\pi}_i \mid, \tag{30}$$

where $\mid \boldsymbol{\pi}_i \mid$ is the sum of all elements included in probability vector $\boldsymbol{\pi}_i$.

The expected number of busy robots, i.e., the expected number of orders being processed in the system, can be obtained by Eq.(31)

$$L_{DC} = \sum_{i+j \leq R} (i+j) \cdot \pi(0,i,j) + \sum_{k=1}^{+\infty} R \cdot \mid \boldsymbol{\pi_k} \mid . \tag{31}$$

The expected number of waiting robots in the buffer area of one workstation can be obtained by

$$L_w = \sum_{i+j \leq R} EL_w(i) \cdot \pi_0(i,j) + \sum_{k=1}^{+\infty} \sum_{i+j=R} EL_w(i) \cdot \pi_k(i,j), \tag{32}$$

where $EL_w(i)$ is the expected number of waiting robots in the buffer area of one workstation when the composite service node $\mu_{c_1}(\cdot)$ has i customers, which can be obtained by the AMVA (see Appendix **B**).

According to Little's Law, the expected waiting time of orders for robots and robots for workstations can be obtained by Eq.(33) and Eq.(34), respectively.

$$W_R = \frac{L_{oq}}{\lambda} \tag{33}$$

$$W_w = \frac{n_w L_w}{\lambda} \tag{34}$$

The system throughput time can be calculated by

$$THT_{DC} = \frac{L_{oq} + L_{DC}}{\lambda}. \tag{35}$$

The utilizations of robots and pickers can be obtained by Eq.(36) and Eq.(37), respectively.

$$\rho_R = \frac{L_{DC}}{R} \cdot 100\%, \tag{36}$$

$$\rho_p = \sum_{i+j \leq R} Qw(i) \cdot \pi_0(i,j) + \sum_{k=1}^{+\infty} \sum_{i+j=R} Qw(i) \cdot \pi_k(i,j), \tag{37}$$

where $Qw(i)$ is the probability that a workstation is busy when the first composite service node has i customers. It can be obtained through the AMVA method (see Appendix **B**).

5. Analytical model validation

In this section, we validate the analytical models by both simulation and a real case.

5.1. Simulation validation

The simulation models are built in Arena (*version* 14.7). The details are presented in Appendix **E**. The system parameters (e.g., bin size, robot velocities and loading/unloading times) are presented in Table 4, and are obtained from a European paper and packaging material distributor in Switzerland, using a robotic-based compact storage and retrieval system.

Table 4: System parameters used in the simulation validation

N	w	l	h_b	R	n_w	v_r	v_l	T_p	t_{lu}	t_t	s	K	τ	γ
5000	0.8m	0.6m	0.33m	30	4	3 m/s	1.6 m/s	$U[5, 15]$	1.2sec	1sec	0.222	500	0.2	0.2

Notes: The product demand D_i follows an ABC curve with skewness parameter $s = 0.222$. That is $D_i = s(i/N)^{s-1}$. The safety stock SS_i is generated from a uniform distribution $U[10, 100]$

Several scenarios are generated by varying the system height H, the order arrival rate λ and the storage and reshuffling policies applied in the system. Specifically, H equals 10 or 15, the width to length ratio r equals one. Under a given storage policy, we can first obtain the required storage capacity C (see Section 3.2), and then the system width W and length L (Eq.(3)). The order arrival rate λ takes 3 levels for each system configuration, corresponding to workstation utilization ranging from 60% to 80%. In total, we have 36 scenarios in the simulation validation (see Table 12 in Appendix **F**). For each scenario, 100 replications are run with a warm-up period of 100 hours and a running time of 1000 hours, leading to a 95% confidence interval where the half-width is less than 2% of the average. Five performance measures are collected and compared with the analytical results, i.e., the system throughput time of dual-command orders THT_{DC}, the expected waiting time of robots for workstations W_w, the utilization of robots ρ_R, the utilization of pickers ρ_p and the filling degree of all storage stacks (denoted by FD, $FD = \frac{\sum_{i=1}^{N} a_i}{L \cdot W \cdot H}$). The accuracy of the analytical models is measured by the absolute relative errors δ,

$$\delta = \frac{\mid A - S \mid}{S} \cdot 100\%$$

where A and S are the analytical and simulation results, respectively.

The average value and the range of δ are presented in Table 5. These results show that the analytical models can estimate the performance of the RCSRS under different storage and reshuffling policies accurately, except for the expected waiting time of robots for workstations W_w. However, note that since W_w is relatively small, a small absolute error will result to a large relative error (see Table 6).

Table 6 presents the average analytical and simulation results, from which we can get the following observations:

1. Compared with random storage stacks, zoning the storage stacks improves the system throughput time with 4.5% for the dedicated storage policy and 7.7% for the shared storage policy, on average.
2. The dedicated storage policy outperforms the shared storage policy with 24.5% on average in terms of system throughput time, while it needs about 51.9% more storage positions. Immediate reshuffling outperforms delayed reshuffling with 12.1% on average in terms of dual command throughput time.

Table 5: Average and range of relative errors of analytical results to simulation results

SP	Zoning	Resh	H, W, L	THT_{DC}			W_w			ρ_R			ρ_p			FD		
				δ	$\underline{\delta}$	$\overline{\delta}$	δ	$\underline{\delta}$	$\overline{\delta}$	δ	$\underline{\delta}$	$\overline{\delta}$	δ	$\underline{\delta}$	$\overline{\delta}$	δ	$\underline{\delta}$	$\overline{\delta}$
Ded	Rand	-	10,134,134	4.28	1.02	8.53	9.23	2.33	24.43	2.51	0.55	4.45	0.10	0.00	0.84	1.67	0.22	2.39
		-	15,113,113	5.11	2.01	9.78	14.23	4.77	25.58	3.46	1.82	6.82	0.11	0.09	0.56	0.87	0.23	3.12
	Zoned	-	10,134,134	1.78	0.13	5.21	12.80	3.07	27.07	2.43	0.95	3.95	0.08	0.01	0.41	2.12	1.50	4.12
		-	15,113,113	4.09	2.09	7.83	13.47	2.19	25.08	3.76	1.95	4.68	0.10	0.06	0.32	1.91	0.50	2.84
Shared	Rand	Imm	10,94,94	3.02	1.05	5.55	8.91	1.25	13.22	2.51	1.30	4.14	0.56	0.12	1.28	1.45	0.32	4.33
			15,76,76	1.36	0.06	4.61	18.64	6.43	34.80	1.61	1.01	3.27	0.36	0.05	1.22	1.87	0.54	3.55
		Del	10,94,94	4.54	1.14	10.18	8.42	3.36	15.82	2.05	1.05	3.88	0.34	0.10	2.41	1.15	0.78	1.43
			15,76,76	0.95	0.39	4.34	18.28	7.20	31.77	0.84	0.32	1.89	0.08	0.06	0.50	1.56	1.10	3.36
	Zoned	Imm	10,95,95	1.69	0.08	4.21	15.12	2.05	21.99	1.44	0.12	3.40	0.14	0.00	0.87	1.97	0.98	3.78
			15,78,78	1.33	0.02	5.72	19.83	5.62	40.30	1.45	0.41	3.59	0.15	0.04	0.46	1.15	0.22	2.54
		Del	10,95,95	1.03	0.15	3.92	11.12	4.69	20.91	1.02	0.44	1.99	0.53	0.09	1.45	2.54	0.78	3.78
			15,78,78	2.34	0.85	5.87	20.62	6.71	37.97	1.17	0.28	4.74	0.14	0.02	2.12	1.17	0.45	2.39

Notes: 'SP' means storage policy, 'Ded' means dedicated storage policy, 'Shared' means shared storage policy, 'Rand' means random storage stacks, 'Zoned' means zoned storage stacks, 'Resh' means reshuffling policy, 'Imm' means immediate reshuffling and 'Del' means delayed reshuffling. $\underline{\delta}$ is the minimum percentage error, $\overline{\delta}$ is the maximum percentage error.

Table 6: Average analytical and simulation results under the dedicated and shared storage policies

SP	Zoning	Resh	H, W, L	λ	THT^A_{DC}	THT^S_{DC}	W^A_w	W^S_w	$\rho^A_R(\%)$	$\rho^S_R(\%)$	$\rho^A_p(\%)$	$\rho^S_p(\%)$	$FD^A(\%)$	$FD^S(\%)$
Ded	Rand	-	10,134,134	650	95.07	99.41	3.49	3.26	57.20	59.68	65.56	65.43	84.57	83.22
		-		750	95.71	102.13	4.80	5.61	66.31	70.50	75.65	75.67	84.57	82.90
		-		850	97.29	108.45	6.34	8.67	75.57	80.34	85.73	85.80	84.57	83.42
		-	15,113,113	550	88.74	90.15	3.70	3.23	45.19	46.03	63.36	63.51	79.28	80.01
		-		650	89.32	92.89	5.40	5.94	53.75	55.83	74.87	74.80	79.28	79.88
		-		750	90.19	100.14	7.56	10.19	62.55	65.72	86.39	86.29	79.28	80.03
	Zoned	-	10,134,134	650	89.29	89.41	3.46	3.06	54.94	53.70	65.56	65.44	84.57	83.32
		-		750	90.71	91.77	4.59	5.23	62.98	63.58	75.65	75.62	84.57	82.30
		-		850	93.23	97.16	6.03	7.81	71.47	74.24	85.73	85.74	84.57	82.82
		-	15,113,113	550	85.63	81.72	3.70	3.27	43.61	41.66	63.36	63.42	79.28	80.31
		-		650	86.11	84.35	5.37	5.49	51.82	50.83	74.87	74.97	79.28	79.68
		-		750	86.24	91.16	7.32	9.77	59.98	62.90	86.39	86.34	79.28	80.03
Shared	Rand	Imm	10,94,94	650	103.44	104.54	3.48	3.10	62.20	63.02	65.56	65.74	98.05	97.65
				750	104.46	107.10	4.74	4.80	72.06	73.72	76.65	75.68	98.05	97.55
				850	107.91	114.25	6.17	7.11	82.05	85.46	85.74	85.84	98.05	97.64
			15,76,76	550	125.52	125.60	3.68	2.73	63.71	63.07	63.36	63.38	99.97	98.10
				650	127.21	128.83	5.23	4.56	75.59	76.72	74.87	74.97	99.97	98.40
				750	137.00	140.88	6.95	6.53	87.63	89.73	86.39	86.37	99.97	98.00
		Del	10,94,94	650	116.51	117.85	3.44	2.97	69.82	65.52	65.56	65.52	98.05	96.78
				750	119.64	122.47	4.61	4.46	80.83	82.50	75.65	75.66	98.05	96.99
				850	138.32	154.00	5.87	6.25	91.94	94.86	85.74	85.68	98.05	97.30
			15,76,76	550	135.38	134.85	3.65	2.77	68.71	68.49	63.36	63.38	99.97	98.40
				650	139.62	139.80	5.11	4.41	81.49	81.92	74.87	74.87	99.97	98.10
				750	179.84	184.15	6.71	6.26	94.40	95.63	86.39	86.26	99.97	98.80
	Zoned	Imm	10,95,95	650	97.42	95.69	3.47	2.86	58.61	57.57	65.56	65.56	99.87	98.43
				750	98.12	98.04	4.78	4.88	67.93	68.01	75.65	75.83	99.87	96.55
				850	100.01	103.29	6.28	8.05	77.39	79.29	85.74	85.65	99.87	98.89
			15,78,78	550	120.92	117.89	3.69	2.63	61.53	60.10	63.36	63.45	98.77	98.10
				650	122.30	120.60	5.27	4.64	73.30	72.18	74.87	74.90	98.77	98.99
				750	128.46	128.44	7.05	7.47	84.69	85.04	86.39	86.25	98.77	96.33
		Del	10,95,95	650	108.19	106.62	3.47	2.87	65.00	64.13	65.56	65.62	99.87	96.35
				750	109.65	109.81	4.69	4.48	75.29	74.96	75.65	75.57	99.87	96.78
				850	115.61	117.32	6.05	6.56	85.69	86.75	85.74	85.81	99.87	99.10
			15,78,78	550	129.41	126.56	3.67	2.66	65.77	64.38	63.36	63.37	98.77	99.22
				650	131.98	129.81	5.18	4.42	78.03	77.19	74.87	74.98	98.77	96.78
				750	148.47	144.00	6.84	6.41	90.42	90.17	86.39	86.20	98.77	97.77

5.2. Real case validation

In this section, we validate the analytical models through a real case. The real system refers to an Autostore system implemented in a sportswear company, headquartered in Oudenaarde, Belgium (see Lalesse (2016)). The system stores about 45,000 bins that are stacked in a grid with size $84 \times 38 \times 14$ (width$\times$ length $\times$ height), and uses 8 robots and 3 workstations to process the picking orders with an arrival rate 600 per hour. Other system parameters include $v_r = 3.51m/s$, $v_l = 1.4m/s$, $w = 0.65m, l = 0.45m, h_b = 0.35m, T_p = 7s, t_{lu} = 1.2s$ and $t_t = 1s$. The system uses totally mixed storage, i.e., a shared storage policy coupled with random storage stacks, and reshuffles the blocking bins immediately. To compare our analytical model with the real system, we adjust the travel time of the robots in the analytical model, considering the acceleration and deceleration effects. Let D be the travel distance, v_{max} be the maximum velocity (either v_r for the robot or v_l for the lifting mechanism) and a be the acceleration and deceleration rate (a_r for the robot and a_l for the lifting mechanism). Based on whether the robot or the lifting mechanism can reach its maximum velocity in the distance D or not, the travel time can be calculated by Eq.(38) (see Tappia et al. (2016))

$$t = \begin{cases} 2v_{max}/a + (D - \frac{v_{max}^2}{a})/v_{max}, & D > \frac{v_{max}^2}{a} \\ 2\sqrt{\frac{D}{a}}, & D \leq \frac{v_{max}^2}{a}. \end{cases} \tag{38}$$

For this system, the robot acceleration and deceleration rate is $a_r = 5m/s^2$ and the lifting mechanism acceleration and deceleration rate is $a_l = 3m/s^2$. Three performance measures are collected to compare the analytical models with the real case, including the throughput capacity per workstation (denoted by TC_w), the expected waiting time of robots for workstations W_w and the utilization of pickers ρ_p.

Table 7: The results of the real case validation

TC_w^{RC}	TC_w^A	δ_{TC_w}	W_w^{RC}	W_w^A	δ_{W_w}	ρ_p^{RC}	ρ_p^A	δ_{ρ_p}
235.12 orders/hour	227.74 orders/hour	3.14%	3.43 s	3.89 s	13.41%	85.11%	87.05%	2.28%

Notes: 'RC' means real case result and 'A' means analytical result.

Table 7 presents the analytical and real case results, it shows that our analytical model can estimate the system performance of the real system with accuracy.

These validation results motivate us to investigate the following questions in Section 6: the optimal system dimensions under different storage policies, the performance comparison of dedicated *vs.* shared and random *vs.* zoned storage policies.

6. Results

This section presents some insights for the system design and storage and operational policies of an RCSRS. Section 6.1 investigates the optimal system dimensions under different storage policies. Section 6.2 compares the performance of the storage policies examined in this study. In Section 6.3, we consider a cost minimization problem for an RCSRS with a system throughput time requirement.

6.1. Optimal system dimensions

We investigate the optimal system dimensions (length, width and height) for an RCSRS with given inventory level. The number of products N, the fractional product bin demand rates $D(i)$ and the setup cost to holding cost ratio K are given and the conditions of EOQ replenishment hold. The objective is to minimize the system throughput time THT_{DC}. So, we have model (M.1).

$$
\begin{aligned}
&min\ THT_{DC}(H, r, P_A, P_B) \\
&s.t. \begin{cases}
D(i) = s(i/N)^{(s-1)}, i = 1, 2, \cdots, N \\
N_{st} \leq L \cdot W \\
L \leq \hat{L}, W \leq \hat{W}, H \leq \hat{H} \\
r = \frac{W}{L} \\
P_A + P_B + P_C = 1, 0 < P_A < 1, 0 < P_B < 1, 0 < P_C < 1 \\
N, R, \lambda, n_w, K, s, \tau \text{ are given,}
\end{cases}
\end{aligned} \tag{M.1}
$$

where the number of workstations n_w satisfies the stability condition $\mu_w n_w > \lambda$. The first equation is the demand of product i, which is assumed to follow an ABC curve with parameter s. The second inequality secures that the floor space (by the number of storage stacks) is adequate for all products, and the third inequality limits the size of the grid.

What should be noted is that the percentage of products included in each class, i.e., P_A, P_B and P_C, are not included in the case of random storage stacks. So the decision variables are the grid size H and r for the case of random storage stacks, but the grid size H and r plus P_A and P_B for the case of zoned storage stacks. We use the following search procedure to solve Model (M.1):

---

1. In the case of random storage stacks, we calculate the storage space required by product i, i.e., a_i by Eq.(1) and then go to step 2. In the case of zoned storage stacks, we vary P_A, P_B and P_C from 0.1 to 0.8 with a stepsize of 0.1 and get all (P_A, P_B, P_C) combinations that satisfy $P_A + P_B + P_C = 1$. For each combination, we obtain the number of products included in each class N_A, N_B, N_C by Eq.(10), calculate a_i and go to step 2.
2. We vary H from 1 to an upper bound, which is $min\{\max_i\{a_i\}, \hat{H}\}$ for the case of random storage stacks and $\hat{H}$ for the case of zoned storage stacks, with a stepsize of one. Note that the upper bound of H in the case of random storage stacks is $min\{\hat{H}, \max_i\{a_i\}\}$ since a larger H will not reduce the number of storage stacks required, but waste storage space. For each H, we calculate the total storage stacks N_{st} and do the following steps:
   (a) In the case of random storage stacks, the width-to-length ratio r takes $\frac{(3n_w+4)l}{(4n_w+8)w}$ (this is proved in Appendix **D**). In the case of zoned storage stacks, we vary r from a lower bound $\underline{r}$ to an upper bound

$\bar{r}$ with a stepsize of 0.1. For each r, $L = \lceil \frac{N_{st}}{r} \rceil$ and $W = \lceil r \cdot L \rceil$, if $W \leq \hat{W}$ and $L \leq \hat{L}$, (H, r) or (H, r, P_A, P_B) is a feasible solution of Model (M.1).

(b) We calculate the system throughput time THT_{DC} of the feasible solution by the *AM-SOQN*.

3. Among all feasible solutions, we find the one with the minimum system throughput time THT^*_{DC}, which is the optimal system scenario (H^*, r^*) or $(H^*, r^*, P^*_A, P^*_B, P^*_C)$.

Several cases are examined to show the optimal system dimensions. The number of products to be stored is $N = 10000$, the ABC curve skewness parameter s takes five levels, $s = 1$(20%/20%), 0.748(20%/30%), 0.431(20%/50%), 0.222(20%/70%), and 0.065(20%/90%). The fraction of the total storage positions reserved for future growth is $\tau = 20\%$, the order arrival rate is $\lambda = 300$ per hour. The system dimension limitation is $\hat{H} = 40, \hat{W} = 250$ and $\hat{L} = 250$. The system has 4 workstations and uses 20 robots, corresponding to a picker utilization larger than 75%. Other system parameters are taken from Table 4. r varies from 0.1 to 2.5 with a stepsize of 0.1. To reflect the effect of storage policy on the usage of storage stacks, we introduce $N_{st,max}$: the number of storage stacks required by the product with the largest required space, i.e., $\max_i\{N^i_{st}\}$. Moreover, we calculate the retrieval throughput time THT_R by Eq.(39) to investigate the trade-off between the immediate and delayed reshuffling policies.

$$THT_R = \begin{cases} W_R + \bar{T}_{d,r} + \bar{T}^D_{lu} + \bar{T}_{r,w} + W_w + \frac{\bar{T}_w}{2}, & \text{Dedicated storage} \\ W_R + \bar{T}_{d,r} + \bar{T}_{re_1} + \bar{T}^{SI}_{lu} + \bar{T}_{re_2} + \bar{T}_{r,w} + W_w + \frac{\bar{T}_w}{2}, & \text{Shared storage, Immediate reshuffling} \\ W_R + \bar{T}_{d,r} + \bar{T}_{re_1} + \bar{T}^{SD}_{lu} + \bar{T}_{r,w} + W_w + \frac{\bar{T}_w}{2}, & \text{Shared storage, Delayed reshuffling} \end{cases} \quad (39)$$

Table 8: Optimal system dimensions in the case of random storage stacks

SP	Resh	s	H^*	W^*	L^*	C	r^*	$N_{st,max}$	THT_{DC}	THT_R	FD (%)
Ded	–	0.065	18	85	170	260100	0.5	42	91.06	56.00	72.38
	–	0.222	27	87	174	408726	0.5	25	101.67	61.91	74.40
	–	0.431	34	85	170	491300	0.5	11	107.80	65.15	73.76
	–	0.748	21	105	210	463050	0.5	6	108.32	66.96	84.46
	–	1	40	71	142	403280	0.5	1	101.38	60.83	98.02
Shared	Imm	0.065	7	87	174	105966	0.5	–	97.57	66.96	99.51
		0.222	7	111	221	171717	0.5	–	114.47	78.05	99.19
		0.431	8	113	226	204304	0.5	–	121.33	83.86	99.35
		0.748	8	118	235	221840	0.5	–	124.73	86.19	98.75
		1	8	118	236	222784	0.5	–	124.91	86.32	99.38
	Del	0.065	7	87	174	105966	0.5	–	115.09	57.75	99.51
		0.222	9	98	195	171990	0.5	–	135.83	68.28	99.37
		0.431	11	97	193	205931	0.5	–	144.07	72.50	98.91
		0.748	11	100	200	220000	0.5	–	147.38	74.13	99.57
		1	11	101	201	223311	0.5	–	148.09	74.50	99.15

Table 9: Optimal system dimensions in the case of zoned storage stacks

SP	Resh	s	H^*	W^*	L^*	C	r^*	P_A^*	P_B^*	P_C^*	THT_{DC}	THT_R	FD (%)
Ded	–	0.065	18	93	155	259470	0.6	0.1	0.4	0.5	87.22	52.31	72.55
	–	0.222	26	97	161	406042	0.6	0.1	0.2	0.7	99.33	59.39	74.89
	–	0.431	32	88	176	495616	0.5	0.1	0.1	0.8	106.66	64.15	73.22
	–	0.748	21	105	210	463050	0.5	0.1	0.5	0.4	107.89	66.50	84.46
	–	1	40	71	142	403280	0.5	-	-	-	100.92	60.83	98.02
Shared	Imm	0.065	6	104	173	107952	0.6	0.6	0.2	0.2	91.02	64.51	99.34
		0.222	7	122	203	173362	0.6	0.8	0.1	0.1	106.85	77.43	99.51
		0.431	7	134	222	208236	0.6	0.8	0.1	0.1	113.74	82.56	99.58
		0.748	8	130	216	224640	0.6	0.7	0.2	0.1	116.85	86.06	99.72
		1	8	131	217	227416	0.6	-	-	-	117.36	86.27	99.26
	Del	0.065	7	112	140	109760	0.8	0.1	0.3	0.6	106.10	55.37	99.34
		0.222	8	124	177	175584	0.7	0.2	0.7	0.1	126.29	66.32	99.31
		0.431	9	128	182	209664	0.7	0.3	0.6	0.1	135.67	71.80	99.48
		0.748	8	140	200	224000	0.7	0.7	0.2	0.1	140.55	74.06	99.12
		1	9	123	205	226935	0.6	-	-	-	141.33	74.27	99.46

Table 8 and Table 9 presents the optimal system dimensions in the case of random and zoned storage stacks, respectively. We make the following observations:

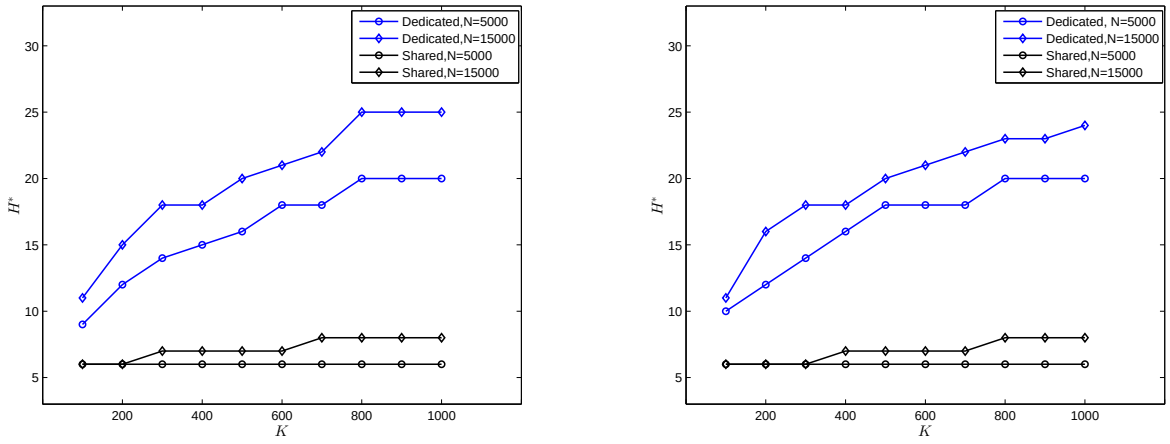
1. For the RCSRS examined in this section, the optimal width-to-length ratio r^* is 0.5 in the case of random storage stacks, corresponding to an optimal width-to-length ratio (in travel time) of $\frac{2}{3}$ (this is proven in Appendix **D**). In the case of zoned storage stacks, the optimal width-to-length ratio r^* is slightly larger than 0.5, which can be explained since zoning storage stacks on x-axis direction will reduce the width travel time.
2. For both random and zoned storage stacks, the total number of storage positions required by the dedicated storage policy is about twice those that are required by the shared storage policy, due to two reasons: the shared storage policy allows space sharing and the filling degree of storage stacks is higher. Moreover, a high grid (with optimal height H^*) fits the dedicated storage policy better, while a flat one (with optimal height H^*) benefits the shared storage policy more. This can be explained as the horizontal robot movement time dominates the vertical lift movement time under the dedicated storage policy, while the vertical lift movement time, especially the reshuffling time, will dominate the horizontal robot movement time under the shared storage policy. Note that H^* should be as high as possible when $s = 1$, i.e., $min\{\hat{H}, \max_i\{a_i\}\}$, to minimize the system throughput time.
3. In terms of dual command throughput time, the dedicated storage policy outperforms the shared storage policy coupled with immediate reshuffling by 10% and zoning storage stacks outperforms random storage stacks by 4.6%, on average. Moreover, compared with delayed reshuffling, immediate reshuffling improves the dual command throughput time by 15.7%, but increases the retrieval throughput time by 13.7%, on average. This means that if an RCSRS is facing high pick demand, it can use delayed reshuffling to improve the retrieval throughput capacity. Otherwise, immediate reshuffling is preferred due to the saving

on dual command throughput time.

6.2. Optimal storage stack height under different storage policies

In Section 6.1, we have analyzed the optimal system dimensions under different storage policies. It appears that the optimal storage stack height H^* varies with the storage policy applied in the system and the characteristics of the storage products, i.e., the skewness parameter s. We also note that product replenishment, in particular, the ratio of ordering cost to holding cost K may also affect the system performance and the optimal system dimensions of the RCSRS. So, we investigate the relationship between the optimal storage stack H^* and K under different storage policies in this section.

We carry out a series of experiments by varying the number of storage products N and the storage policy applied. Specifically, $N = 5000$ or 15000, and both the dedicated and shared storage policies and both random and zoning storage stacks are examined. The product demand follows an ABC curve with skewness parameter $s = 0.065$. For the shared storage policy, we use the immediate reshuffling strategy. K varies from 100 to 1000 with a stepsize of 100 and other system parameters are the same as Section 6.1. We use Model (M.1) to derive the optimal storage stacks height H^* and present the results in Fig.10.



(a) Random storage stacks

(b) Zoned storage stacks

Figure 10: Optimal storage stack height

The results show that the optimal storage stack height H^* increases significantly with the ratio of setup cost to holding cost K under the dedicated storage policy, while it increases only slightly under the shared storage policy. This means that if the replenishment cost goes up, the replenishment period will become longer and the replenishment quantity will become larger based on the EOQ. As a result, the RCSRS should use a higher grid to increase the inventory level in the storage stacks. Conversely, when the replenishment period shortens and the replenishment quantity reduces, the RCSRS should lower the grid to reduce the inventory level in the storage stacks. Compared with the dedicated storage policy, the optimal storage stack height under the shared storage policy is relatively smaller and much less sensitive to K. This means that a high grid fits the dedicated storage policy better, while a low grid is more beneficial for the shared storage policy. The optimal storage

stack height obtained for the shared storage policy is much more robust to changes in K than that obtained for the dedicated storage policy.

6.3. Cost minimization models

The results of Section 6.1 show that the dedicated storage policy outperforms the shared storage policy in terms of the system throughput time THT_{DC}, which can be explained since the reshuffling of blocking bins under the shared storage policy increases the system throughput time, while no reshuffling is required under the dedicated storage policy. However, despite the disadvantage of a shared storage policy on system throughput time, it can save cost since less storage space is required due to space sharing. Therefore, the selection between these two storage policies is actually a trade-off between the operational efficiency and the system costs. In this section, we study the cost minimization problem in RCSRSs under both the dedicated and shared storage policies.

The number of products stored in the system (N) and their demands (D_i) are given. The order arrival rate λ is known and the number of workstations n_w is the minimum number that satisfies the stability condition $n_w \cdot \mu_w > \lambda$. The objective is to design the system to minimize the system total annual cost TC, with the constraint that the system throughput time THT_{DC} should be less than a critical level $THT_{DC_{max}}$. The system cost consists of three parts: the cost of robots, the cost of storage positions and the cost of floor space. Since the number of workstations is fixed, we exclude the picking labor cost from the objective in model (M.2). The results of Section 6.2 show that zoning storage stacks can improve the system throughput time, compared with random storage stacks. So, to minimize the system total cost, we use the optimal zoning method under both the dedicated and shared storage policies. This results in the total annual cost minimization model (M.2)

$$
\begin{aligned}
&min\ TC(H, r, R, P_A, P_B) = C_R \cdot R + C_{SP} \cdot L \cdot W + C_{FS} \cdot L \cdot W \cdot H \\
&s.t. \begin{cases}
N_{st} \le L \cdot W \\
THT_{DC}(L, W, H, R) \le THT_{DC_{max}} \\
L \le \hat{L}, W \le \hat{W}, H \le \hat{H} \\
D_i = s(i/N)^{(s-1)}, i = 1, 2, \cdots, N \\
r = \frac{W}{L} \\
P_A + P_B + P_C = 1, 0 < P_A < 1, 0 < P_B < 1, 0 < P_C < 1 \\
N, \lambda, n_w, K, \tau \text{ are given}
\end{cases}
\end{aligned}
\tag{M.2}
$$

where C_R is the annualized cost per robot, C_{SP} is the annual cost per storage position and C_{FS} is the annual cost per square meter floor space. The first constraint is to ensure that the total number of storage stacks is adequate for all products. The second constraint is to ensure that the system throughput time of the RCSRS is smaller than the maximum throughput time required, i.e., $THT_{DC} \le THT_{DC_{max}}$. The third constraint is the limitation on the size of the grid. The decision variables are the number of robots R, the percentage of products included in class A and B, i.e., P_A, P_B, and the grid size H and r.

We use a grid search procedure to solve the Model (M.2):

---

1. We vary P_A, P_B and P_C from 0.1 to 0.8 with a stepsize of 0.1 and obtain all (P_A, P_B, P_C) combinations that satisfy $P_A + P_B + P_C = 1$. For each combination, we calculate the storage space required by product i, i.e., a_i by Eq.(1) and then go to step 2.
2. We vary H from 1 to an upper bound, which is $min\{\max_i\{a_i\}, \hat{H}\}$ for the dedicated storage policy and $\hat{H}$ for the shared storage policy, with a stepsize of one. For each H, we get the total storage stacks N_{st} by Eq.(4) for the dedicated storage policy and by Eq.(7) for the shared storage policy, and then do the following steps:
   (a) We vary r from a lower bound $\underline{r}$ to an upper bound $\overline{r}$ with a stepsize of 0.1. For each r, we take $L = \lceil \frac{N_{st}}{r} \rceil$ and $W = \lceil r \cdot L \rceil$. If $W \leq \hat{W}$ and $L \leq \hat{L}$, go to step b.
   (b) We find the minimum number of robots R that satisfies $THT_{DC}(H, r, R, P_A, P_B) \leq THT_{DC_{max}}$ by the *AM-SOQN*, and then record the (H, r, R, P_A, P_B) as a feasible solution of Model (M.2).
3. We calculate the total annual cost of all feasible solutions, and select the one with the minimum total annual cost.

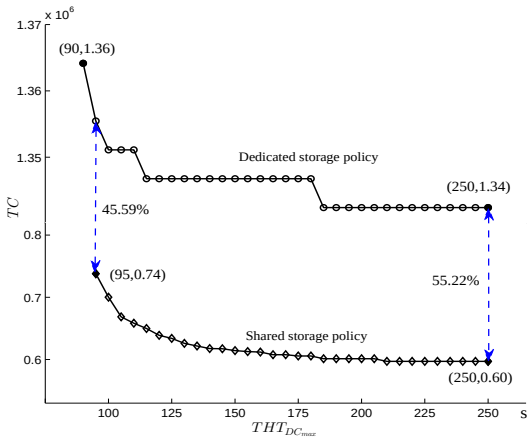
As an example, we consider an RCSRS that needs to store $N = 10,000$ products whose demand follow ABC curves with skewness parameters $s = 0.065, 0.222$ or 0.576. The investment cost per robot is € 30,000 (annualization in 7 years), the investment cost per storage position is € 40 (annualization in 10 years) and the investment cost of floor space in a warehouse per square meter is € 500 (annualization in 30 years). We consider an interest rate $IR = 0.5\%$. Then, the annual costs of a robot, a storage position and a square meter warehouse floor space are

$$C_R = \sum_{t=1}^{7} \frac{30000(1+IR)^{t-1}}{7}, C_{SP} = \sum_{t=1}^{10} \frac{40(1+IR)^{t-1}}{10}, C_{FS} = \sum_{t=1}^{30} \frac{500(1+IR)^{t-1}}{30}$$

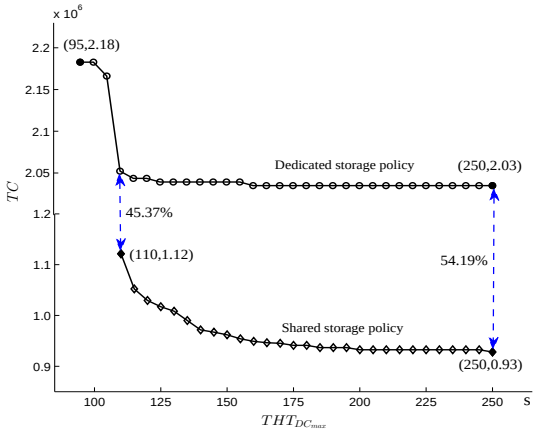
The picking order arrival rate is $\lambda = 300$ per hour and other system parameters come from Table 4. We solve Model (M.2) and present the trade-off curves between optimal system cost and the required system throughput time in Fig.11. The optimal system scenarios (system dimensions W^*, L^*, H^* and number of robots R^*) for each required system throughput time $THT_{DC_{max}}$ are included in Table 13 in Appendix **G**.

Fig.11 shows that, for a given maximum system throughput time $THT_{DC_{max}}$, the shared storage policy can substantially reduce the system cost of an RCSRS, compared with the dedicated storage policy. The optimal annualized system costs offered by the shared storage policy are almost 50% (on average) lower than that offered by the dedicated storage policy, i.e. the costs under dedicated storage are twice as large as that under shared storage. The reasons are twofold: compared with the shared storage policy, much larger storage space is required by the dedicated storage policy and the filling degree of storage stacks is relatively lower (see Table 13)

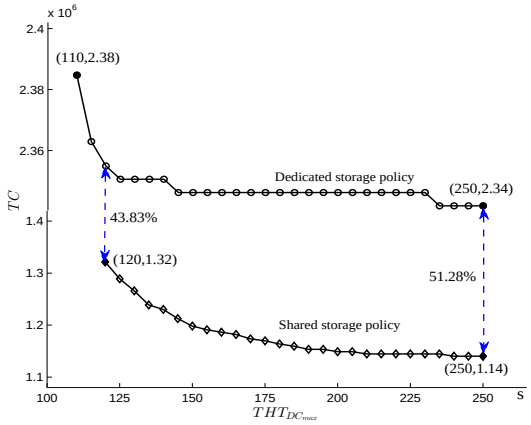
Except for the advantage of the shared storage policy on system cost, Fig.11 also reflects the advantage of the dedicated storage policy on the system throughput time. The dedicated storage policy can offer a system



(a) $s = 0.065$



(b) $s = 0.222$



(c) $s = 0.576$

Figure 11: Optimal cost of RCSRS with throughput time constraint

throughput time shorter than that of the shared storage policy. Specifically, by using the dedicated storage policy, the system throughput time THT_{DC} can be between $[90, 95)$ when $s = 0.065$, $[95, 110)$ when $s = 0.222$ and $[110, 120)$ when $s = 0.576$. Therefore, the trade-off between these two storage policies is that the shared storage policy can save the system cost substantially, while the dedicated storage policy can offer a very short system throughput time that the shared storage policy can not achieve.

7. Conclusions and Further work

This study considers a new material-handling technology: robot-based compact storage and retrieval systems (RCSRS). In such a system, the inventory items are stored in standard plastic bins, and the roof is a grid formed by rectangular cells under which bins are stored on top of each other. Robots with transporting and lifting functionalities are used to transport bins on the grid. Several workstations are located at the lowest level of several side storage stacks, where goods enter and leave the system. With high storage space utilization and outstanding throughput capacity, RCSRSs have seen many implementations in recent years, particularly in e-commerce retailers.

This study focuses on performance estimation and storage policy evaluation of RCSRS. We consider both dedicated and shared storage policies per storage stack, and random and zoned storage stacks within the grid. Under the shared storage policy, we propose immediate and delayed reshuffling strategies for the reshuffling process that is required before getting a bin in the deep level. Semi-open queueing networks are built to estimate the performance of RCSRS under various storage policies and reshuffling policies, including the system throughput time, the expected waiting time of orders for robots, the expected waiting time of robots for workstations and the utilizations of robots and workstations. We first approximate the original models by reduced SOQNs with two load-dependent service nodes, and then use Matrix-Geometric Method to solve them. Both simulations and a real case are used to validate the analytical models. The relative errors show that the analytical models can estimate the system performance with accuracy.

We built system dimensions optimization models to minimize the system throughput time. The results show that the optimal ratio of width-to-length in traveling time is around $\frac{2}{3}$ in the case of random storage stacks, and slightly larger in the case of zoned storage stack. A high grid (with an optimal height) fits the dedicated storage policy better, while a flat grid (with an optimal height) is more beneficial for the shared storage policy. We also compare the performance of the immediate and delayed reshuffling policies and found that the immediate reshuffling policy can improve the dual command throughput time by sacrificing the retrieval throughput time. To decide between the dedicated and shared storage policies, we also consider the cost minimization problem of the RCSRSs with a requirement on system throughput time. We examined a series of cases and found that the shared storage policy dominates the dedicated storage policy in terms of system cost, due to its substantial saving on total storage space, while the dedicated storage policy can offer a short system throughput time that the shared storage policy can not.

For future studies, it is interesting to apply uni-directional routes on the grid to avoid congestion and blocking of robots. The assignment rules of orders to robots and robots to workstations can also be investigated, as it may improve the system performance.

Acknowledgement

This research is partially supported by the National Natural Science Foundation of China (grant number 71131004), (grant number 71471071), and Chutian Scholarship of Hubei Province of China.

References

Cai, X., Heragu, S. S., & Liu, Y. (2014). Modeling and evaluating the avs/rs with tier-to-tier vehicles using a semi-open queueing network. *IIE Transactions*, *46*, 905–927.

CIMCORP (2016). http://cimcorp.com/, .

De Koster, R., Le-Duc, T., & Yu, Y. (2008). Optimal storage rack design for a 3-dimensional compact as/rs. *International journal of production research*, *46*, 1495–1514.

ElementLogic (2016a). http://www.elementlogic.net/en/portfolio-items/also-actebis/, .

ElementLogic (2016b). http://www.elementlogic.net/en/portfolio-items/bergans/, .

ElementLogic (2016c). http://www.elementlogic.net/en/xxl-i-orebro/, .

Fukunari, M., & Malmborg, C. (2008). An efficient cycle time model for autonomous vehicle storage and retrieval systems. *International Journal of Production Research*, *46*, 3167–3184.

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.

Gharehgozli, A. H., Laporte, G., Yu, Y., & De Koster, R. (2015). Scheduling twin yard cranes in a container block. *Transportation Science*, *49*, 686–705.

Gharehgozli, A. H., Yu, Y., De Koster, R., & Udding, J. T. (2014a). A decision-tree stacking heuristic minimising the expected number of reshuffles at a container terminal. *International Journal of Production Research*, *52*, 2592–2611.

Gharehgozli, A. H., Yu, Y., De Koster, R., & Udding, J. T. (2014b). An exact method for scheduling a yard crane. *European Journal of Operational Research*, *235*, 431–447.

Gharehgozli, A. H., Yu, Y., Zhang, X., & De Koster, R. (2014c). Polynomial time algorithms to minimize total travel time in a two-depot automated storage/retrieval system. *Transportation Science*, .

Graves, S. C., Hausman, W. H., & Schwarz, L. B. (1977). Storage-retrieval interleaving in automatic warehousing systems. *Management Science*, *23*, 935–945.

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

Hao, J., Yu, Y., & Zhang, L. L. (2015). Optimal design of a 3d compact storage system with the i/o port at the lower mid-point of the storage rack. *International Journal of Production Research*, *53*, 5153–5173.

Hatteland (2016). http://hatteland.com/en/, .

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.

Jia, J., & Heragu, S. S. (2009). Solving semi-open queuing networks. *Operations research*, *57*, 391–401.

Kleinrock, L. (1976). Queuing systems. *Theory, Wiley, New York-London-Sydney-Toronto*, *1*.

Kota, V. R., Taylor, D., & Gue, K. R. (2015). Retrieval time performance in puzzle-based storage systems. *Journal of Manufacturing Technology Management*, *26*, 582–602.

Lalesse, E. (2016). Autostore order pick system. *https://www.youtube.com/watch?v=7DeT-Hj2DXk*, .

Lerher, T. (2015). Travel time model for double-deep shuttle-based storage and retrieval systems. *International Journal of Production Research*, (pp. 1–22).

Lerher, T., Ekren, B. Y., Dukic, G., & Rosi, B. (2015). Travel time model for shuttle-based storage and retrieval systems. *The International Journal of Advanced Manufacturing Technology*, *78*, 1705–1725.

Malmborg, C. J. (2002). Conceptualizing tools for autonomous vehicle storage and retrieval systems. *International Journal of Production Research*, *40*, 1807–1822.

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.

Roy, D., Krishnamurthy, A., Heragu, S., & Malmborg, C. (2015). Queuing models to analyze dwell-point and cross-aisle location in autonomous vehicle-based warehouse systems. *European Journal of Operational Research*, *242*, 72–87.

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.

Swisslog (2016a). http://swisslog.com/en/success-stories/wds/references/antalis, .

Swisslog (2016b). http://swisslog.com/en/success-stories/wds/references/asda, .

Tappia, E., Roy, D., De Koster, R., & Melacini, M. (2016). Modeling, analysis, and design insights for shuttle-based compact storage systems. *Transportation Science in press*, .

Vis, I. F., & Carlo, H. J. (2010). Sequencing two cooperating automated stacking cranes in a container terminal. *Transportation Science*, *44*, 169–182.

Yang, P., Miao, L., Xue, Z., & Qin, L. (2015). Optimal storage rack design for a multi-deep compact as/rs considering the acceleration/deceleration of the storage and retrieval machine. *International Journal of Production Research*, *53*, 929–943.

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

Yu, Y., & De Koster, R. (2009b). Optimal zone boundaries for two-class-based compact three-dimensional automated storage and retrieval systems. *IIE Transactions*, *41*, 194–208.

Yu, Y., De Koster, R., & Guo, X. (2015). Class-based storage with a finite number of items: Using more classes is not always better. *Production and Operations Management*, *24*, 1235–1247.

Zaerpour, N., Yu, Y., & De Koster, R. (2015a). Small is beautiful: A framework for evaluating and optimizing live-cube compact storage systems. *Transportation Science*, .

Zaerpour, N., Yu, Y., & De Koster, R. (2015b). Storing fresh produce for fast retrieval in an automated compact cross-dock system. *Production and Operations Management*, *24*, 1266–1284.

Appendix A. Service time expressions in the SOQN

The mean value of the movement time $T_{d,r}$ (under both dedicated and mixed storage policies) can be calculated by the following equation.

$$\bar{T}_{d,r} = \sum_{x_d=1}^{W} \sum_{y_d=1}^{L} \sum_{x_r=1}^{W} \sum_{y_r=1}^{L} [\frac{\mid x_d - x_r \mid \cdot w + \mid y_d - y_r \mid \cdot l}{v_r} + g \cdot t_t] \cdot P(x_d)P(y_d)P(x_r)P(y_r)$$

We can obtain the mean values of both movement times T_{r,w_i} and $T_{w_i,s}$ by the same logic. The resulting first two moments of service times of the service nodes in the SOQN are presented in Table 10

Appendix B. Approximate Mean Value Method (AMVA)

Table 11 presents the notations used in the AMVA

The AMVA method includes the following steps:

1. Initialize. Let $p_m(0 \mid 0) = 1, Q_m(0) = 0, EL_m(0) = 0, m = 1, 2, \cdots, M$.
2. Preprocessing. Enumerate n from 0 to N, do the following procedures:
   (a) For $m = 1, 2, \cdots, M$, calculate

$$ET_m(n) = Q_m(n-1)ES_{rem,m} + EL_m(n-1)\frac{ES_m}{c_m} + ES_m,$$

   where $ES^r_{rem,m}(\vec{n})$ is given by the following equation

$$ES_{rem,m} = \frac{c_m - 1}{c_m + 1} \cdot \frac{ES_m}{c_m} + \frac{2}{c_m + 1} \cdot \frac{1}{c_m} \cdot \frac{ES_m^2}{2ES_m}.$$

   (b) Calculate the load-dependent throughput

$$TH(n) = \frac{n}{\sum_{m=1}^{M} v_m ET_m(n)},$$

   (c) For $m = 1, 2, \cdots, M$ and $l = 1, 2, \cdots, min(c_m - 1, n)$, calculate

$$p_m(l \mid n) = \frac{ES_m(n)}{l} v_m TH(n) p_m(l-1 \mid n-1).$$

   (d) For $m = 1, 2, \cdots, M$, if $n < c_m, Q_m(n) = 0$, otherwise,

$$Q_m(n) = \frac{ES_m}{c_m} v_m TH(n) \cdot [Q_m(n-1) + p_m(c_m - 1 \mid n-1)].$$

Table 10: The first two moments of service times of service nodes in the SOQN

Service node	Storage policy	Mean value of service time	scv of service time
$\mu_{d,r}$	random stacks	$\frac{Ww+Ll}{3v_r} + \frac{(LW-W-L-3)t_t}{LW-1}$	
	zoned stacks	$[\frac{P_A^2 w_A}{3} + P_B^2(\frac{w_B}{3} + \frac{w_A}{2}) + P_C^2(\frac{w_C}{3} + \frac{w_B}{2} + \frac{w_A}{2}) + 2P_AP_B(w_A + \frac{w_B}{2}) + 2P_AP_C(w_A + w_B + \frac{w_C}{2}) + P_BP_C(w_A + w_B + \frac{w_C}{2})]\frac{w}{v_r} + \frac{Ll}{3} + \frac{(LW-W-L-3)t_t}{LW-1}$	$cv_{d,r}^2 = \frac{E[T_{d,r}^2] - \bar{T}_{d,r}^2}{\bar{T}_{d,r}}$
μ_{r,w_i}	random stacks	$\frac{Ww}{2v_r} + \frac{1}{L}\sum_{j=1}^{L}\frac{\lvert j-y_{w_i}\rvert l}{v_r} + \frac{t_t}{L}$	$cv_{r,w_i}^2 = \frac{E[T_{r,w_i}^2] - \bar{T}_{r,w_i}^2}{\bar{T}_{r,w_i}}$
	zoned stacks	$\frac{Ww}{2v_r} + \frac{1}{L}\sum_{j=1}^{L}\frac{\lvert j-y_{w_i}\rvert l}{v_r} + \frac{t_t}{L}$	
$\mu_{w_i,s}$	random stacks	$\frac{Ww}{2v_r} + \frac{1}{L}\sum_{j=1}^{L}\frac{\lvert j-y_{w_i}\rvert l}{v_r} + \frac{t_t}{L}$	
	zoned, dedicated	$\frac{Ww}{2v_r} + \frac{1}{L}\sum_{j=1}^{L}\frac{\lvert j-y_{w_i}\rvert l}{v_r} + \frac{t_t}{L}$	
	zoned, shared	$[P_A(\frac{w_A}{4} + \frac{w_B}{2} + \frac{w_C}{2}) + P_B(\frac{w_B}{4} + \frac{w_C}{2}) + \frac{P_C w_C}{4}]\frac{w}{v_r} + \sum_{j=1}^{L}\frac{\lvert j-y_{w_i}\rvert l}{v_r} + \frac{t_t}{L}$	$cv_{w_i,s}^2 = \frac{E[T_{w_i,s}^2] - \bar{T}_{w_i,s}^2}{\bar{T}_{w_i,s}}$
μ_w	-	$\frac{1}{2(\frac{Hh_b}{v_l} + t_{lu}) + \frac{\lambda(a+b)}{2n_w}\cdot\frac{(b-a)^2+6(a+b)}{24}}$	$\frac{\lambda^2(b-a)^2}{12\lambda n_w + \frac{1152 n_w^2 (\frac{Hh_b}{v_l} + t_{lu})}{(a+b)[6(a+b)+(b-a)^2]}}$
μ_{re_1}	shared	$[(\frac{w\tau^2}{9v_r} - \frac{2h_b\tau^2}{9v_l} + \frac{h_b\tau}{2v_l})H^2 + (\frac{w\tau}{6v_r} - \frac{h_b}{2v_l} + \frac{h_b\tau}{6v_l})H + \frac{h_b}{6v_l} - \frac{w}{3v_r}](1+\gamma) + \frac{t_{lu}\tau H}{2} - \frac{t_{lu}}{2}$	$cv_{re_1}^2 = \frac{E[T_{re_1}^2] - \bar{T}_{re_1}^2}{\bar{T}_{re_1}}$
μ_{re_2}	shared	$[(\frac{w\tau^2}{9v_r} - \frac{2h_b\tau^2}{9v_l} + \frac{h_b\tau}{2v_l})H^2 + (\frac{w\tau}{6v_r} - \frac{h_b}{2v_l} + \frac{2h_b\tau}{3v_l})H + \frac{h_b}{6v_l} - \frac{w}{3v_r}](1+\gamma) + \frac{t_{lu}\tau H}{2} - \frac{t_{lu}}{2}$	$cv_{re_2}^2 = \frac{E[T_{re_2}^2] - \bar{T}_{re_2}^2}{\bar{T}_{re_2}}$
μ_{lu}^D	dedicated	$\sum_{i=1}^{N}\frac{p_i}{I_i}\sum_{h=1}^{I_i}\frac{2(H-h)h_b}{v_l} + t_{lu}$	
μ_{lu}^S	shared	$[\frac{(2-\tau)h_bH}{v_l} - \frac{h_b}{v_l}](1+\gamma) + t_{lu}$	$cv_{lu}^2 = \frac{E[T_{lu}^2] - \bar{T}_{lu}^2}{\bar{T}_{lu}}$
μ_{lu}^{SI}	shared, immediate	$[\frac{(2-\tau)h_bH}{v_l} - \frac{h_b}{v_l}](1+\gamma) + 3t_{lu} + \frac{3l}{v_r}$	
μ_{lu}^{SD}	shared, delay	$[\frac{(2-\tau)h_bH}{v_l} - \frac{h_b}{v_l}](1+\gamma) + t_{lu}$	

Notes: The parameter γ models the honeycombing effect, which allows inflating the travel time in the storage stack.

(e) For $m = 1, 2, \cdots, M$, calculate

$$p_m(0 \mid n) = 1 - \sum_{l=1}^{min(c_m-1,n)} p_m(l \mid n) - Q_m(n).$$

(f) For $m = 1, 2, \cdots, M$ and $r = 1, 2$, if $n < c_m$, $EL_m(n) = 0$, otherwise,

$$EL_m(n) = \frac{ES_m(n)}{c_m} v_m TH(n)[EL_m(n-1) + Q_m(n-1)].$$

Table 11: Notations used in AMVA

Notations	Meaning
M	Number of service stations in the closed queueing network.
c_m	the number of servers in service station m. For infinite server, we let c_m be an infinite number.
v_m	the visit ratio of customer at service station m.
N	the number of customers in the closed queueing network.
$ES_{rem,m}$	the expected time remaining until the first departure of customer at service station m.
ES_m	the service time of service station m (the first moment).
ES_m^2	the squared coefficient of variation of service station m (the second moment).
$p_m(l \mid n)$	the probability that there are l customers in service station m when the system contains n customers.
$Q_m(n)$	the probability that all servers at service station m are busy when the system contains n_s storage transactions and n_r retrieval transactions.
$EL_m(n)$	the mean number of customers in the queue of service station m (excluding jobs in service) when the system contains n customers.
$ET_m(n)$	the lead time of service station m when the system contains n customers.
$TH(n)$	the system throughput when the system contains n customers.

Appendix C. Matrix-Geometric Method

We specify the details of the generator matrix $\mathbf{Q}$. $\mathbf{B}_{00}$ is the transition matrix from state $\boldsymbol{\pi_0}$ to state $\boldsymbol{\pi_0}$,

$$\mathbf{B}_{00} = \begin{bmatrix} -\lambda & \lambda & & & & \\ \begin{bmatrix} 0 \\ \mu_{c_2}(1) \end{bmatrix} & \begin{bmatrix} -\lambda-\mu_{c_1}(1) & \mu_{c_1}(1) \\ 0 & -\lambda-\mu_{c_2}(1) \end{bmatrix} & \begin{bmatrix} \lambda & 0 & 0 \\ 0 & \lambda & 0 \end{bmatrix} & & & \\ & \begin{bmatrix} 0 & 0 \\ \mu_{c_2}(1) & 0 \\ 0 & \mu_{c_2}(2) \end{bmatrix} & \begin{bmatrix} -\lambda-\mu_{c_1}(2) & \mu_{c_1}(2) & 0 \\ 0 & -\lambda-\mu_{c_1}(1)-\mu_{c_2}(1) & \mu_{c_1}(1) \\ 0 & 0 & -\lambda-\mu_{c_2}(2) \end{bmatrix} & \begin{bmatrix} \lambda & 0 & 0 & 0 \\ 0 & \lambda & 0 & 0 \\ 0 & 0 & \lambda & 0 \end{bmatrix} & & \\ & \ddots & \ddots & \ddots & & \\ & & \begin{bmatrix} 0 & \cdots & 0 \\ \mu_{c_2}(1) & 0 & \vdots \\ & \ddots & \vdots \\ & & \mu_{c_2}(R) \end{bmatrix} & \begin{bmatrix} -\lambda-\mu_{c_1}(R) & 0 & \cdots & 0 \\ 0 & -\lambda-\mu_{c_1}(R-1)-\mu_{c_2}(1) & 0 & \vdots \\ & & \ddots & \vdots \\ & & & -\lambda-\mu_{c_2}(R) \end{bmatrix} \end{bmatrix}.$$

$\mathbf{B}_{01}$ is the transition matrix from state $\boldsymbol{\pi_0}$ to state $\boldsymbol{\pi_1}$,

$$\mathbf{B}_{01} = \begin{bmatrix} \mathbf{0}_{\frac{R(R+1)}{2}\times(R+1)} \\ \hline \lambda & & \\ & \ddots & \\ & & \lambda \end{bmatrix}_{\frac{(R+1)(R+2)}{2}\times(R+1)}.$$

$\mathbf{B_{10}}$ is the transition matrix from state $\boldsymbol{\pi_1}$ to state $\boldsymbol{\pi_0}$,

$$\mathbf{B}_{10} = \left[\begin{array}{ccccc} & 0 & & & \\ \mathbf{0}_{(R+1)\times\frac{R(R+1)}{2}} & \mu_{c_2}(1) & 0 & & \\ & & \ddots & \ddots & \\ & & & \mu_{c_2}(R-1) & 0 \\ & & & & \mu_{c_2}(R) \quad 0 \end{array}\right]_{(R+1)\times\frac{(R+1)(R+2)}{2}}.$$

$\mathbf{A}$ is the transition matrix from state π_i to state $\pi_{i+1}, i \geq 1$,

$$\mathbf{A} = \begin{bmatrix} \lambda & & \\ & \ddots & \\ & & \lambda \end{bmatrix}_{(R+1)\times(R+1)}.$$

$\mathbf{B}$ is the transition matrix from state π_i to state $\pi_i, i \geq 1$,

$$\mathbf{B} = \begin{bmatrix} -\lambda-\mu_{c_1}(R) & \mu_{c_1}(R) & & & & \\ & -\lambda-\mu_{c_1}(R-1)-\mu_{c_2}(1) & \mu_{c_1}(R-1) & & & \\ & & \ddots & & & \\ & & & -\lambda-\mu_{c_1}(1)-\mu_{c_2}(R-1) & \mu_{c_1}(1) & \\ & & & & -\lambda-\mu_{c_2}(R) & \end{bmatrix}_{(R+1)\times(R+1)}.$$

$\mathbf{C}$ is the transition matrix from state π_{i+1} to state $\pi_i, i \geq 1$,

$$\mathbf{C} = \begin{bmatrix} 0 & & & & & \\ \mu_{c_2}(1) & 0 & & & & \\ & \ddots & \ddots & & & \\ & & \mu_{c_2}(R-1) & 0 & & \\ & & & \mu_{c_2}(R) & 0 & \end{bmatrix}_{(R+1)\times(R+1)}.$$

The repetitive structure of the generator matrix $\mathbf{Q}$ secures that the following equation holds for steady state probability $\boldsymbol{\pi}_i$ (Neuts 1980)

$$\boldsymbol{\pi}_{i+1} = \boldsymbol{\pi}_i \mathbf{R}, i \geq 1,$$

where $\mathbf{R}$ is the transition rate matrix and

$$\boldsymbol{\pi}_1 \mathbf{R}^i \cdot \mathbf{A} + \boldsymbol{\pi}_1 \mathbf{R}^{i+1} \cdot \mathbf{B} + \boldsymbol{\pi}_1 \mathbf{R}^{i+2} \cdot \mathbf{C} = 0, \quad i \geq 1.$$

After some simplification, we get

$$\mathbf{R} = -(\mathbf{A} + \mathbf{R}^2\mathbf{C})\mathbf{B}^{-1}. \tag{40}$$

Based on Eq (40), we can obtain $\mathbf{R}$ by the following iterative method (Tappia et al. (2016), Cai et al. (2014))

$\mathbf{R}_0 = \mathbf{0}$, $\mathbf{R}_1 = -(\mathbf{A} + \mathbf{R}_0^2\mathbf{C})\mathbf{B}^{-1}$, $k = 0$

while $|| \mathbf{R}_{k+1} | - | \mathbf{R}_k ||> \varepsilon$

$k++$;

$\mathbf{R}_{k+1} = -(\mathbf{A} + \mathbf{R}_k^2\mathbf{C})\mathbf{B}^{-1}$;

end

$\mathbf{R}$=$\mathbf{R}_k$

Appendix D. Proof of optimal width to length ratio in the case of random storage stacks

The objective of optimizing the width to length ratio (in traveling time) is to minimize the expected movement time of robots on the grid in a dual command cycle, i.e., $T = \bar{T}_{d,r} + \bar{T}_{r,w} + \bar{T}_{w,s}$. In the case of random

storage stacks, both storage and retrieval positions could be in any cell with the same probability. Moreover, any workstation will be equally likely assigned to the robot. So, we can write T as Eq.(D.1)

$$T = \frac{1}{v_r}[\frac{W \cdot w}{3} + \frac{L \cdot l}{3} + 2(\frac{W \cdot w}{2} + \frac{1}{n}\sum_{i=1}^{n}\frac{1}{L}\sum_{j=1}^{L} \mid j - y_{w_i} \mid \cdot l)], \tag{D.1}$$

where $n = \frac{n_w}{2}$, y_{w_i} is the y-axis coordinate of the ith workstation and $y_{w_i} = \frac{iL}{n+1}$.

After some simplification, we get

$$T = \frac{1}{v_r}[\frac{4Ww}{3} + \frac{(3n+2)Ll}{3n+3}] \geq \frac{2}{v_r}\sqrt{\frac{4Ww}{3} \cdot \frac{(3n+2)Ll}{3n+3}} = \frac{2}{v_r}\sqrt{\frac{4(3n_w+2)wl\frac{C}{H}}{3(3n+3)}}.$$

Therefor, T takes the minimum value when $\frac{4Ww}{3} = \frac{(3n+2)Ll}{3n+3}$. So, the optimal width to length ratio (in traveling time) is $r_t^* = \frac{W \cdot w}{L \cdot l} = \frac{(3n_w+4)}{(4n_w+8)}$. In a RCSRS with 4 workstations, $r_t^* = \frac{2}{3}$ and $r^* = \frac{1}{2}$.

Appendix E. Simulation model

The simulation model contains the following processes (see Fig.12 for a flow diagram):

1. The orders arrive at the system, following a Poisson process with mean inter-arrival time $\frac{1}{\lambda}$.
2. The system randomly assigns an available robot to the order. If all robots are busy, the order will wait in the order queue.
3. The seized robot moves from its dwell point to the retrieval point according to a shortest path. In the case of random storage stacks, both the robot dwell point and the retrieval point are uniformly distributed over the stacks. In the case of zoned storage stacks, they will be in each zone with a probability proportional to the percentage of the turnover of products in that zone.
4. The robot fetches the retrieval bin. This process may include reshuffling of blocking bins. Under the dedicated storage policy, the robot directly picks up the top bin. Under the shared storage policy, the system will first identify whether the retrieval bin is on the top of the storage stack. If not, the robot will first reshuffle the blocking bins on the grid and then fetch the retrieval bin. The blocking bins will be returned according to the original stack sequence. If immediate reshuffling is used, the robot will put the retrieval bin on the grid temporarily, and then return the blocking bins into the stack. If delayed reshuffling is used, returning of the blocking bins will be done afterwards.
5. The robot transports the retrieval bin to the designated workstation according to a shortest path. There, the robot drops off the retrieval bin and picks up a storage bin.
6. The robot transports the storage bin to the storage point. In the case of random stacks, the stack that will hold the storage bin will be located equally likely at any position. In the case of zoned stacks, the probability that the robot will go to one zone is determined by the turnover of products in it.
7. The robot drops off the bin on the top of the storage stack.
8. If the previous retrieval operation includes reshuffling of blocking bins and the delayed reshuffling rule is used, the robot will move back to the retrieval point and returns the blocking bins into the storage stack.

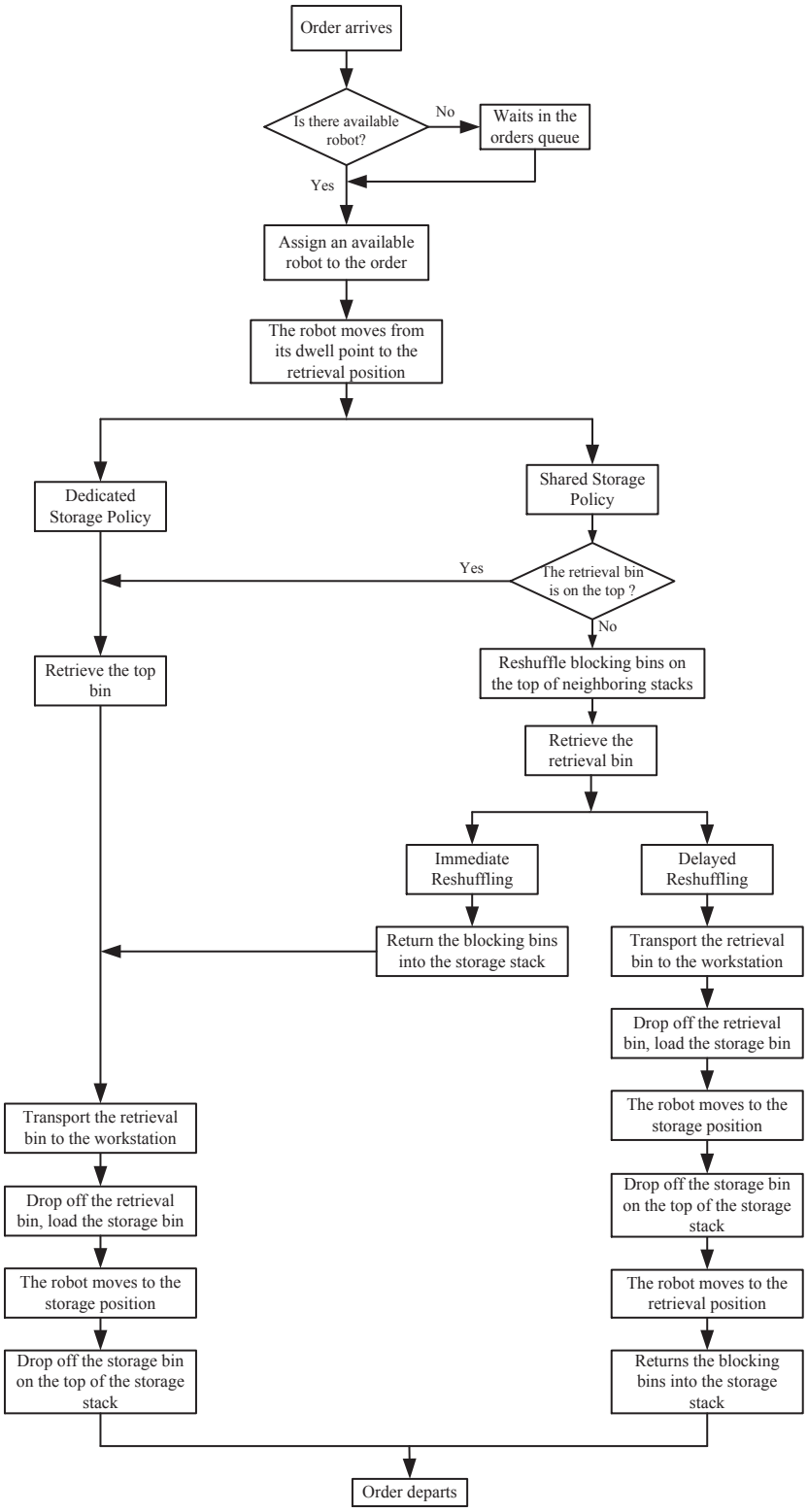


Figure 12: Flowchart of simulation models

Appendix F. System scenarios for simulation validation

Table 12: System scenarios examined in simulation validation

Scenario	H	W	L	R	horizontal	vertical	reshuffle	zone	λ (per hour)
1-3	10	134	134	30	Random	Dedicated	–	–	650,750,850
4-6	15	113	113	30	Random	Dedicated	–	–	550,650,750
7-9	10	134	134	30	Zoned	Dedicated	–	[0.6,0.3,0.1]	650,750,850
10-12	15	113	113	30	Zoned	Dedicated	–	[0.6,0.3,0.1]	550,650,750
13-15	10	94	94	30	Random	Shared	Immediate	–	650,750,850
16-18	15	76	76	30	Random	Shared	Immediate	–	550,650,750
19-21	10	94	94	30	Random	Shared	Delayed	–	650,750,850
22-24	15	76	76	30	Random	Shared	Delayed	–	550,650,750
25-27	10	95	95	30	Zoned	Shared	Immediate	[0.6,0.3,0.1]	650,750,850
28-30	15	78	78	30	Zoned	Shared	Immediate	[0.6,0.3,0.1]	550,650,750
31-33	10	95	95	30	Zoned	Shared	Delayed	[0.6,0.3,0.1]	650,750,850
34-36	15	78	78	30	Zoned	Shared	Delayed	[0.6,0.3,0.1]	550,650,750

Notes: S mean Scenarios. R takes different values for dedicated and shared storage policies to ensure that $\rho_R < 1$.

Appendix G. Optimal system scenarios to minimize the total system cost

Table 13: The optimal system scenarios to minimize the total costs

			$THT_{DC_{max}}$																
SP	s		90	100	110	120	130	140	150	160	170	180	190	200	210	220	230	240	250
		W^*	112	112	112	112	112	112	112	112	112	112	112	112	112	112	112	112	112
		L^*	140	140	140	140	140	140	140	140	140	140	140	140	140	140	140	140	140
		H^*	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16	16
	0.065	R^*	13	10	10	9	9	9	9	9	9	9	8	8	8	8	8	8	8
		P_A^*	0.2	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
		P_B^*	0.7	0.6	0.1	0.4	0.1	0.1	0.1	0.1	0.1	0.1	0.8	0.7	0.6	0.6	0.5	0.4	0.4
		P_C^*	0.1	0.3	0.8	0.5	0.8	0.8	0.8	0.8	0.8	0.8	0.1	0.2	0.3	0.3	0.4	0.5	0.5
		W^*	NaN	153	140	140	140	140	140	140	140	140	140	140	140	140	140	140	140
		L^*	NaN	153	200	200	200	200	200	200	200	200	200	200	200	200	200	200	200
		H^*	NaN	18	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13
Ded	0.222	R^*	NaN	9	14	12	11	11	11	10	10	10	10	10	10	10	10	10	10
		P_A^*	NaN	0.2	0.1	0.1	0.1	0.1	0.1	0.4	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
		P_B^*	NaN	0.6	0.5	0.1	0.3	0.1	0.1	0.5	0.5	0.2	0.1	0.1	0.1	0.1	0.1	0.1	0.1
		P_C^*	NaN	0.2	0.4	0.8	0.6	0.8	0.8	0.1	0.4	0.7	0.8	0.8	0.8	0.8	0.8	0.8	0.8
		W^*	NaN	NaN	121	155	155	155	155	155	155	155	155	155	155	155	155	155	155
		L^*	NaN	NaN	200	155	155	155	155	155	155	155	155	155	155	155	155	155	155
		H^*	NaN	NaN	19	19	19	19	19	19	19	19	19	19	19	19	19	19	19
	0.576	R^*	NaN	NaN	16	13	12	12	11	11	11	11	11	11	11	11	11	10	10
		P_A^*	NaN	NaN	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
		P_B^*	NaN	NaN	0.3	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.4	0.2
		P_C^*	NaN	NaN	0.6	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.5	0.7
		W^*	NaN	87	67	68	82	58	55	63	63	64	64	64	64	64	64	64	64
		L^*	NaN	124	134	113	82	116	109	90	90	80	80	80	80	80	80	80	80
		H^*	NaN	10	12	14	16	16	18	19	19	21	21	21	21	21	21	21	21
	0.065	R^*	NaN	15	13	14	15	13	15	16	15	17	16	16	15	15	15	15	15
		P_A^*	NaN	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
		P_B^*	NaN	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.1	0.1	0.1	0.1
		P_C^*	NaN	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.7	0.8	0.8	0.8	0.8
		W^*	NaN	NaN	98	85	79	104	101	98	62	93	93	93	93	93	93	93	93
		L^*	NaN	NaN	196	170	158	104	101	98	155	93	93	93	93	93	93	93	93
		H^*	NaN	NaN	9	12	14	16	17	18	18	20	20	20	20	20	20	20	20
Shared	0.222	R^*	NaN	NaN	16	14	16	16	16	16	15	18	17	16	16	16	16	16	15
		P_A^*	NaN	NaN	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
		P_B^*	NaN	NaN	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
		P_C^*	NaN	NaN	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8
		W^*	NaN	NaN	NaN	118	116	88	80	110	107	72	89	89	89	89	89	89	89
		L^*	NaN	NaN	NaN	168	145	176	160	110	107	144	111	111	111	111	111	111	111
		H^*	NaN	NaN	NaN	11	13	14	17	18	19	21	22	22	22	22	22	22	22
	0.576	R^*	NaN	NaN	NaN	17	16	15	18	18	18	20	20	19	18	18	18	17	17
		P_A^*	NaN	NaN	NaN	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
		P_B^*	NaN	NaN	NaN	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
		P_C^*	NaN	NaN	NaN	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8	0.8