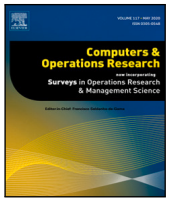
Computers & Operations Research 163 (2024) 106505

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor



Scheduling of automated guided vehicles for tandem quay cranes in automated container terminals

Lingrui Kong, Mingjun Ji *, Anxu Yu, Zhendi Gao

Department of Transportation Engineering, Dalian Maritime University, Linghai Road No. 1th, Ganjingzi District, Dalian 116026, Liaoning Province, China

ARTICLE INFO

Keywords:
Automated container terminal
Tandem quay crane
Traffic congestion and conflicts
Limited yard buffers

ABSTRACT

This study investigates the automated guided vehicle scheduling problem for serving the tandem quay cranes in the automated container terminal. The tandem quay crane is equipped with two spreaders, allowing it to execute both single-lift and tandem-lift operations. When performing a tandem-lift, two automated guided vehicles are required to support the tandem quay crane's operation. Considering the coordination between tandem quay cranes and automated guided vehicles, a mixed-integer linear programming model is established to minimize the completion time of unloading operations by the tandem quay cranes. The formulation takes into account various crucial factors, such as traffic congestion and conflicts among automated guided vehicles, and the capacity limitation of yard buffers. A multi-start local search algorithm is developed for solving the problem. Computational experiments demonstrate the algorithm's efficiency and effectiveness in solving both small- and large-scale instances. Furthermore, the computational analysis highlights the importance of considering traffic congestion and conflicts, as well as limited yard buffers in the automated container terminal. The study also reveals that when a sufficient number of vehicles are available, the deployment of tandem quay cranes can lead to a substantial enhancement of approximately 30% in operational efficiency compared to conventional single-spreader quay cranes.

1. Introduction

Over 80% of global trade is carried by sea Guo et al. (2023) and global container traffic has experienced rapid growth in the past two decades (Liu and Ge, 2018). In 2021, the total throughput of the world's container terminals reached 857 million TEUs (UNCTAD, 2022). The increase in container throughput as well as the trend towards large-scale vessels pose challenges to container terminals. In response to these challenges and to gain competitiveness, ports have been gradually transitioning towards the development of automated container terminals or semi-automated container terminals (Zhong et al., 2019). Such a shift is motivated by the pursuit of sustainable development and the significant benefits that automation brings. Automated equipment, such as automated guided vehicles (AGVs) and automated cranes, has been widely adopted in these terminals due to their intelligent driving capabilities, low probability of operational errors, and high efficiency in loading and unloading operations.

Furthermore, to enhance terminal productivity and reduce ship turnaround time, novel equipment with higher operational efficiency has been developed. One such innovation is the tandem quay crane (TQC), a specialized type of machinery designed for loading and unloading containers onto and from containerships at the quayside. Unlike conventional quay cranes (CQCs) with a single spreader, the TQC is equipped with two spreaders, allowing it to lift two 40 ft containers or four 20 ft containers simultaneously, which is known as a tandem-lift. Additionally, the TQC can perform a single-lift by using just one of its spreaders to lift one 40 ft container or two 20 ft containers (as illustrated in Fig. 1).

To fully utilize the designed efficiency of the TQC, it is essential to prioritize tandem-lift operations as much as possible. Since an AGV can only transport one 40 ft container or two 20 ft containers at most in a single trip, two AGVs are required to serve a tandem-lift of the TQC. As a result, proper organization of AGVs plays a crucial role in ensuring the TQC's optimal performance. Consider the following scenario as an example: when two AGVs arrive at the TQC simultaneously (as shown in Fig. 1(a)), it is convenient for the TQC to execute a tandem-lift. However, as illustrated in Fig. 2, in situations where one AGV arrives first, a decision must be made whether the TQC should perform a single-lift for the arrived AGV (allowing the first AGV to leave) or wait for the other AGV to execute a tandem-lift (then both AGVs can depart). The above analysis highlights the significant impact of AGV arrival times on the operations of the TQC. Hence, an appropriate AGV schedule is vital to ensure the operational efficiency of the TQC.

* Corresponding author.
E-mail address: jmj@dlmu.edu.cn (M. Ji).

https://doi.org/10.1016/j.cor.2023.106505
Received 9 March 2023; Received in revised form 24 October 2023; Accepted 1 December 2023
Available online 2 December 2023
0305-0548/© 2023 Elsevier Ltd. All rights reserved.

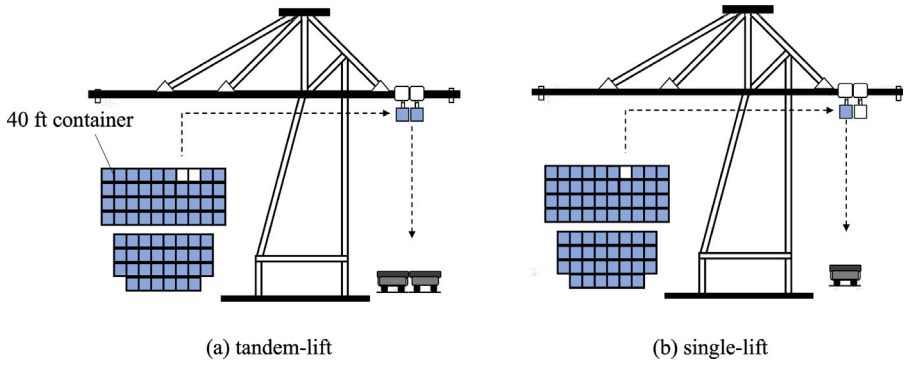


Fig. 1. Illustrations of the tandem-lift and single-lift.

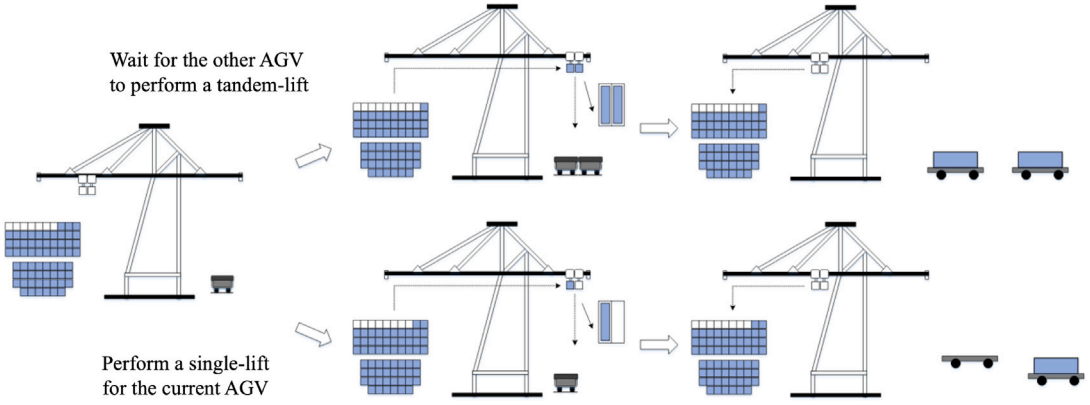


Fig. 2. TQC decisions when only one AGV arrives.

A typical layout of an automated container terminal (ACT) is illustrated in Fig. 3, where three types of handling equipment operate closely together. The quay cranes (QCs) are responsible for loading and unloading containers onto and from the containership at the quayside; the AGVs are responsible for transporting containers between the quayside and the yard area; and the automated stacking cranes (ASCs) are responsible for container stacking and retrieval in the yard. At the seaside (landside) end of each yard block are transfer buffers where the container transfer between the AGV (external truck) and the ASC is conducted. To reduce congestion and deadlocks caused by direct interaction between the AGV and the ASC, AGV-mates are implemented at transfer buffers at the seaside end of each yard block. The unloading process in the ACT concerning the TQC and AGV-mates is depicted in Fig. 4, which can be described as follows: a TQC discharges the container(s) onto an AGV (or two AGVs), which will transport the container to the transfer buffer at the seaside end of the assigned block in the yard. The related AGV-mate unloads the container from the AGV, after which the AGV can leave the yard; the ASC collects the container from the AGV-mate and moves the container from the AGV-mate to the corresponding storage location in the block. The loading process is the reverse of the unloading process.

It is important to highlight that the transfer buffers (AGV-mates) located at the seaside end of each block have limited capacity. If an AGV arrives at a block where all transfer buffers are already occupied, the AGV must wait for an available buffer. The waiting time of AGVs at the yard is directly influenced by the number of transfer buffers at the seaside end of the block. Consequently, the AGV waiting time becomes a critical factor that cannot be ignored, especially when the number of transfer buffers is restricted. Therefore, when scheduling AGVs, the limited number of buffers at the seaside end of each block should be taken into consideration.

Moreover, in an ACT with a perpendicular stack layout (as shown in Fig. 3), to alleviate the traffic congestion and conflicts during AGV transportation, unidirectional travel paths are commonly adopted, and multiple perpendicular shortcuts are utilized to minimize AGV travel time between the quayside and the yard area (Roy et al., 2020). Despite the implementation of these measures, AGVs remain susceptible to conflicts and congestion during their transportation. This susceptibility is mainly attributed to the simultaneous operations of multiple AGVs within the ACT. Therefore, when scheduling AGVs in the ACT, the traffic congestion and conflicts among AGVs should not be ignored. In addressing traffic congestion, previous research commonly adopted an approach by restricting that at most one AGV could be present on any path segment simultaneously (i.e., different AGVs cannot occupy the same path at the same time). This type of approach may be overly conservative and fail to fully utilize the path capacity. When the path capacity exceeds one, such models and methods become inapplicable.

Based on the aforementioned analysis, this study focuses on the unloading process at ACTs and investigates the AGV scheduling problem for serving the TQCs, considering the traffic congestion and conflicts among AGVs, as well as the limited capacity of yard buffers. The aim of this study is to enhance the operational efficiency of TQCs utilized in ACTs, and the contributions can be summarized as follows. First, we establish a model to schedule AGVs for serving TQCs in ACTs, effectively managing the interactions among TQCs, AGVs, AGV-mates, and ASCs. Second, we handle the issue of traffic congestion and conflicts by introducing new variables and constraints that better handle the capacity characteristics of AGV path segments. Third, we design a multi-start local search algorithm equipped with high-quality searching rules, which is capable of providing approximate solutions for large-scale instances within a reasonable time.

The rest of this paper is structured as follows. Section 2 offers a review of relevant literature on equipment scheduling problems at container terminals related to TQCs and AGVs. In Section 3, we elaborate on the AGV scheduling problem in ACTs, where TQCs are utilized, and

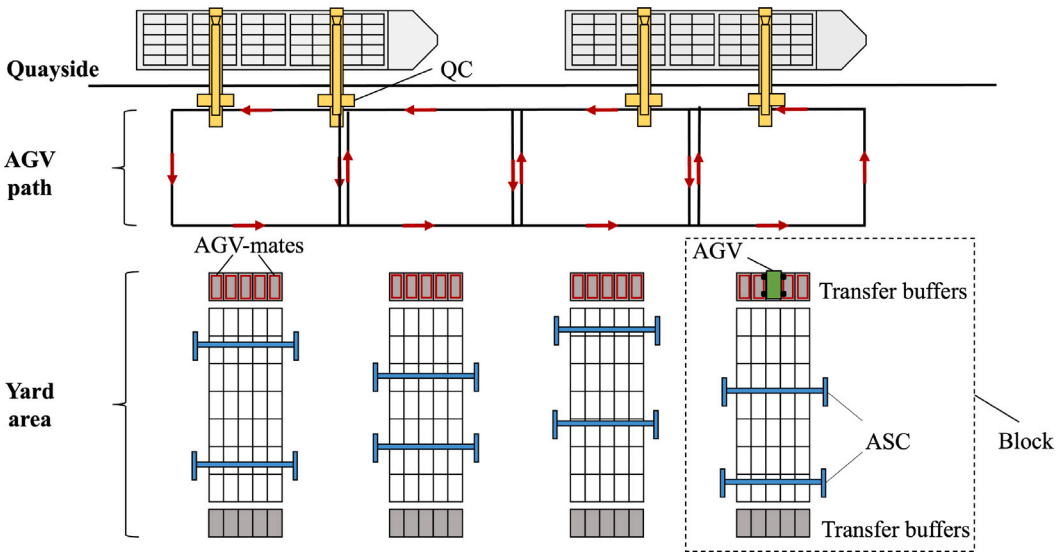


Fig. 3. A typical layout at the ACT.

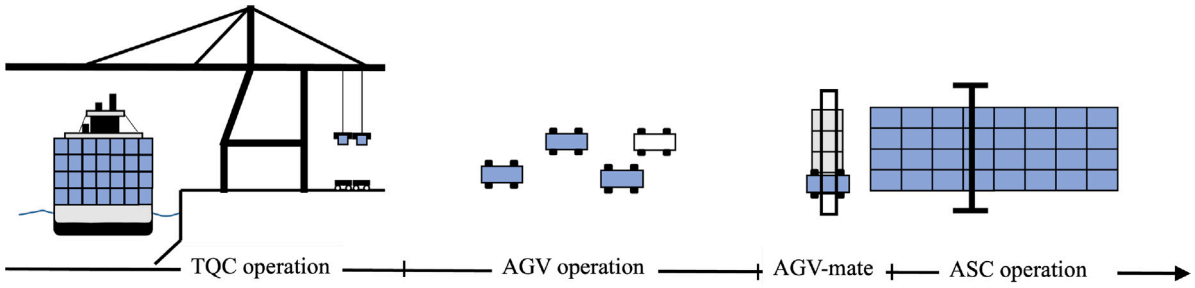


Fig. 4. The unloading process in the ACT.

present the mathematical model. Subsequently, Section 4 introduces the multi-start local search algorithm designed to solve the problem. In Section 5, we conduct experiments to verify the efficiency of the proposed algorithm and demonstrate the significance of our research. Finally, Section 6 summarizes the main findings of this study, and identifies potential areas for future research.

2. Literature review

Considerable research has been done on the equipment scheduling problems at container terminals. Here, we provide an overview of the existing literature related to the TQC and introduce the former research on the AGV scheduling problem in ACTs.

Extensive research has been conducted to address the QC scheduling problems in the container terminal. Interested readers can find investigations and classifications of the QC scheduling problems in the reviews by Bierwirth and Meisel (2010, 2015), and Boysen et al. (2017). For the purpose of this discussion, we narrow our focus to studies associated with TQCs. Notably, studies related to the TQC mainly focuses on the introduction of the equipment itself (such as Chao and Lin, 2011; Bartošek and Marek, 2013; Choi et al., 2014), the TQC-related scheduling problem has garnered limited attention. Huang and Li (2017), Lashkari et al. (2017), and Kong et al. (2022) focused on the scheduling problem of the TQC by considering its characteristics and restrictions; both heuristic and exact algorithms were proposed for solving the TQC scheduling problem. However, these studies did not account for the interdependencies between TQC schedules and those of other equipment. Xing et al. (2012) and Chen et al. (2014) designed approaches for scheduling AGVs and yard trucks, respectively, while accommodating TQC operations. Kong et al. (2021) further analyzed other related schedules (i.e., container slot plans) that affect the loading operations of the TQC, and developed an integrated optimization model for slot planning and truck scheduling according to the characteristics of the TQC. However, the above studies all failed to address the interactions between the horizontal transport equipment and the handling equipment in the yard. The yard often emerges as a bottleneck in container terminals, particularly in ACTs with perpendicular layouts. Consequently, the waiting time of the horizontal transport equipment at the yard should not be ignored and the interactions between the transport equipment and the yard should be well confined in the ACT.

AGVs are important transportation vehicles that transport containers between the QC on the quayside and the ASC on the yard side. The efficiency of its operations directly impacts the effectiveness of both QCs and ASCs. Due to its significance, scholars have conducted a series of studies to address the AGV scheduling problem in ACTs (such as Kim and Bae, 2004; Grunow et al., 2006; Angeloudis and Bell, 2010; Bian et al., 2015; Xin et al., 2015; Choe et al., 2016; Xu et al., 2020; Zheng et al., 2022; Drungilas et al., 2023). Considering the impact of container storage plans on the AGV scheduling problem, Luo and Wu (2015), Luo et al. (2016), and Hu et al. (2019) addressed the joint vehicle scheduling and storage allocation problem in ACTs.

Due to the interactions between AGVs and other automated equipment in ACTs, some researchers have investigated the collaborative optimization issues between AGVs and other automated equipment. Meersmans and Wagelmans (2001) first investigated the integrated scheduling of AGVs and ASCs. Lau and Zhao (2008) established an MIP formulation for the integrated scheduling of intelligent handling equipment and presented a multi-layer generic algorithm. Xin et al. (2014) proposed a method of dispatching QCs, AGVs, and YCs to

improve the operational performance of an ACT by combining the handling capacity and energy consumption objectives. Chen et al. (2020) studied the integrated rail-mounted yard crane and AGV scheduling problem as a multi-robot coordination and scheduling problem. Xin et al. (2022) investigated the integrated scheduling problem of QCs and AGVs by considering the energy and efficiency issues. Xing et al. (2023) conducted research on AGV dispatching and equipment scheduling with speed optimization. The above research was carried out without considering AGV-mates. Yang et al. (2018) and Zhong et al. (2019, 2020) addressed the concept of the AGV-mates in the ACT; however, the capacity of yard buffers (AGV-mates) was assumed to be infinite, which is not consistent with reality. To be more practical, Nguyen and Kim (2009), Sadeghian et al. (2014), Jonker et al. (2021), Zhang et al. (2021), Zhuang et al. (2022), and Duan et al. (2023) considered the yard buffers with limited capacity. Yin et al. (2023) addressed the integrated scheduling problem of QCs and shuttle vehicles by considering limited apron buffer capacity.

Although extensive studies have been conducted on AGV scheduling problems as well as integrated scheduling problems in ACTs, little attention has been given to the congestion and conflicts among vehicles. Yang et al. (2018) formulated a bi-level programming model for an integrated scheduling problem. The upper-level model is the integrated scheduling problem of QCs, AGVs, and YCs. The lower-level model is AGV path planning by considering the path capacity of no more than two. Tommaso et al. (2018) addressed the conflict-free pickup and delivery problem to determine the vehicle paths and speeds on each arc of the path, with AGV scheduling in ACTs being one area that appeared in this problem. They restricted that at most one vehicle can traverse an arc of the transportation network at any time. Chen et al. (2020) investigated the yard crane and AGV scheduling problem based on the extended time-space network that modeled the AGV congestions, however, the potential conflicts among vehicle movements were not taken into consideration. Zhong et al. (2020) studied the integrated scheduling of multi-AGVs with conflict-free path planning with cranes to minimize AGV delay time. They addressed the issues of both AGV congestion and conflicts, i.e., deadlocks at intersections of the traffic network; however, they also tackled the AGV congestion issues by confining a fixed path capacity of no more than two and assumed that the task allocation is known in advance. Wang and Zeng (2022) investigated the AGV dispatching and routing problem with multiple bidirectional paths to generate a conflict-free route, and a tailored branch-and-bound algorithm was developed to solve the problem within a reasonable amount of time. They handled conflict issues by avoiding the opposite direction conflict of different AGVs on the same horizontal path. Zhong et al. (2023) addressed the joint scheduling problem of AGVs and YCs aiming to minimize the energy consumption, while restricting that the path capacity of AGVs is no more than two. Liu et al. (2023) focused on the integrated scheduling of horizontal transport equipment and handling equipment, as well as path planning for AGVs in a sea-rail intermodal container terminal with a U-shaped yard layout. They tackle the AGV congestion issue by assuming that each arc can only be occupied by one AGV at the same time. In summary, the preceding literature that addressed the congestion challenges of AGVs in the ACT handled congestion by limiting path capacity to just one or two. However, in specific scenarios where path segments are long or wide, the segment's capacity exceeds this limitation. Under such circumstances, the aforementioned approaches become unsuitable. Furthermore, the previous models established to handle AGV congestion and conflicts within the ACT were nonlinear, making them complex to comprehend and challenging to resolve using programming solvers.

Based on the analysis of the aforementioned literature (for a more comprehensive survey of AGV-based vehicle transportation, interested readers can refer to Sun et al. (2022)), it can be inferred that despite the considerable research on scheduling intelligent handling equipment in container terminals, there is a notable gap in addressing scheduling issues related to TQCs. Furthermore, the challenge of AGV congestion has been inadequately addressed, often overlooking the actual capacity of traffic paths. In order to enhance TQC efficiency and facilitate seamless coordination among handling equipment in the ACT, this study seeks to offer comprehensive solutions to the AGV scheduling problem, particularly in servicing TQCs. By effectively managing AGV congestion, resolving conflicts, and harmonizing interactions among handling equipment, this research strives to significantly enhance the overall operational efficiency at the ACT.

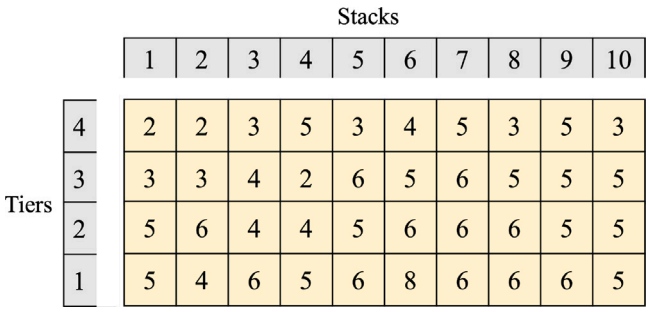


Fig. 5. Illustration of an even bay that loaded with 40 ft containers.

3. Problem formulation

3.1. Problem description

In this study, we address the container unloading process concerning TQCs, AGVs, AGV-mates, and ASCs at the ACT with a perpendicular layout.

For TQC operations, both single-lifts and tandem-lifts are permissible. However, given the unique attributes of the TQC, containers involved in a tandem-lift must be sourced from two adjacent stacks within the same tier of an even bay, and the total weight of the lifted containers must be less than a specific threshold. Consequently, not all containers are conducive to tandem-lifts. For instance, when the total weight of two adjacent 40 ft containers surpasses the TQC's stipulated limit, the tandem-lift is prohibited for these containers. An illustration of an even bay loaded with 40 ft containers is presented in Fig. 5, with the weight of each corresponding container indicated in the middle of the slot. If the weight limitation of the tandem-lift is 10, the container in Tier 1 and Stack 3 cannot be lifted together with the container in Tier 1 and Stack 4 in a tandem-lift, as their total weight reaches 11. Specifically, in this context, we define that to execute a tandem-lift for a 40 ft container implies that the container and its immediate right-adjacent 40 ft container are lifted simultaneously. Given that container positions and weight details are known before the unloading process begins, we can mark the containers that are available for tandem-lifts.

As depicted in Fig. 3, AGVs travel in a counterclockwise circulation pattern, which helps mitigate conflicts and collisions to a certain extent. In this study, both AGV congestion and conflicts are taken into consideration. To address congestion concerns, the number of AGVs simultaneously present on each path segment should not exceed a specific threshold, known as the path capacity. Regarding conflicts, potential traffic conflicts, as illustrated in Fig. 6, encompass opposite conflict, node conflict, and occupancy conflict. In the counterclockwise AGV circulation, there are no opposite conflicts, and adjustments to AGV speeds can prevent occupancy conflicts. The path capacity constraint also contributes to reducing occupancy conflicts among AGVs. Therefore, the focus of traffic conflicts in this study primarily revolves around node conflicts, which arise when two or more AGVs are unable to traverse the same path intersection simultaneously.

For the unloading process, the storage positions of the containers on the containership and in the yard are known. The AGV path map

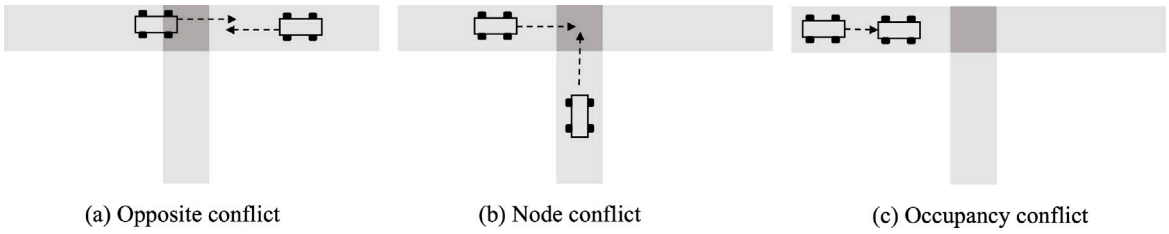


Fig. 6. Traffic conflicts of AGVs.

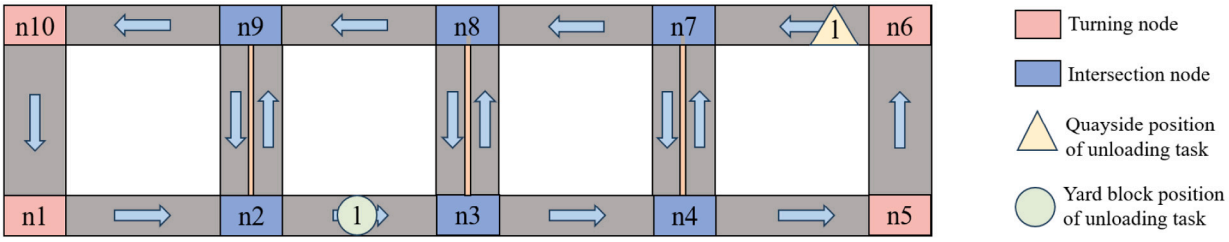


Fig. 7. Illustration of AGV path map.

is established in Fig. 7, which is set according to the AGV transport network. In Fig. 7, the pink rectangulars represent the turning nodes, i.e., n1, n5, n6, n10; the blue rectangulars represent intersection nodes, where potential node conflicts might arise, i.e., n2, n3, n4, n7, n8, n9; the yellow triangle denotes the quayside position for a specific unloading task (container 1); and the green circle symbolizes the yard block position for the same unloading task (container 1).

In the yard area, several transfer buffers are implemented at the seaside end of each yard block, each transfer buffer is equipped with an AGV-mate to avoid congestion and deadlocks caused by direct interaction between the AGV and the ASC. The yard buffers are considered to be limited.

With the above descriptions, we formulate a model with the objective of minimizing the completion time (makespan) for unloading operations performed by the TQCs. To achieve this goal, the model encompasses the determination of operational modes for TQCs (whether to execute a tandem-lift or single-lift for each container), job assignments for the AGVs, the sequencing of containers for each AGV, and the timing of actions carried out by TQCs, AGVs, AGV-mates, and ASCs for each container. The model takes into account various factors, including the operational sequence of TQCs, limited transfer buffers (AGV-mates), congestion and conflict constraints among AGVs, as well as blocking constraints between AGVs and TQCs/AGV-mates, all of which play crucial roles in shaping the overall operational framework.

3.2. Assumptions

(1) All containers are of a 40 ft size. Thus, the TQC is capable of unloading one container at a time during a single-lift, and two containers during a tandem-lift. Similarly, ASC operates with one container at a time, and AGV transports a single container at a time as well. Consequently, a tandem-lift requires the implementation of two AGVs.

(2) The allocation and schedule plan of the quay cranes is provided, as it is typically established during the initial phases along with the berth allocation plan. Consequently, we assume that the requirement for non-interference and safety margin between TQCs has already been met. For the unloading operations in a bay, the policy "from-land-to-sea" with "tier-by-tier" is applied by each TQC.

(3) The storage positions of the containers on the containership and in the yard are already known. The AGV always chooses the path with the shortest travel distance between the quayside and the yard (to minimize both energy assumption and travel time). For instance, to transport container 1 as depicted in Fig. 7 from its quayside position to its assigned yard block, the AGV should pass nodes n7, n8, n9, and n2, sequentially.

(4) All the AGVs are identical, and the time for an AGV spent on the AGV-mate is assumed to be a fixed value. The TQC/ASC processing time for a container is related to the storage position of that container in the containership/yard block. The duration of TQC movement between bays has already been factored into the TQC processing time parameter for a specific container.

(5) Two ASCs are deployed within a yard block, and congestion between them is unavoidable. Consequently, we estimate an extra ASC processing time for each container, accounting for the congestion between two ASCs. This calculation is closely tied to the storage positions of containers within the yard block and the arrival rates of AGVs and external trucks at the block. For more detailed information, please refer to Appendix.

(6) We discretize the time horizon and adopt the assumption that the time taken for AGVs to navigate through an intersection (or turning) node is equivalent to one unit of time.

3.3. Mathematical formulation

The mathematical notations needed in the model development are listed as follows.

Sets

Q: Set of TQCs with unloading tasks; the TQCs are sequentially numbered from left to right along the quayside.

Ω: Set of unloading containers; the containers are numbered according to the series number and the unloading sequence of TQCs (an example can be seen in Fig. 8).

B: Set of blocks in the yard.

Φ: Set that contains the last task of each TQC; i.e., $\Phi = \{12, 30\}$ in Fig. 8.

θ: Set that contains containers that allow the tandem-lift operations. If $i \in \theta$, it indicates that containers i and $i+1$ can be unloaded simultaneously in a tandem-lift by the TQC.

P: Set that contains all the travel path segments of AGVs. Each path segment is identified by a number, as exemplified in Fig. 9,

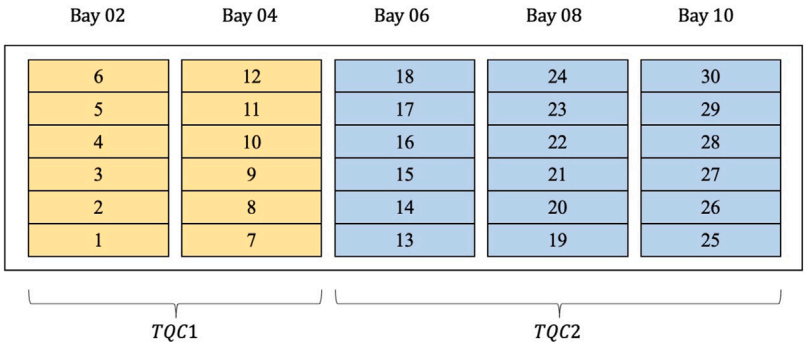


Fig. 8. Illustration of series number of containers.

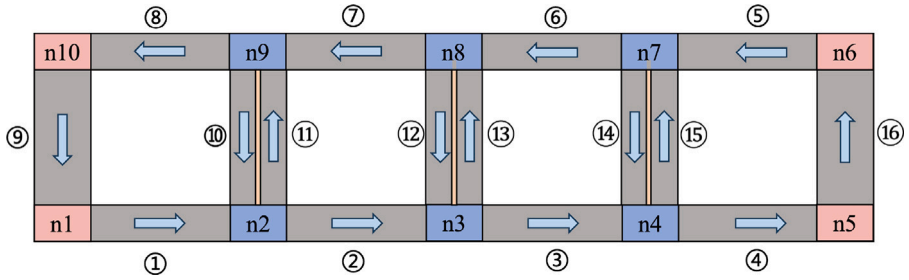


Fig. 9. Illustration of series number of path segments.

where the numbers within circles indicate the sequential numbering of corresponding path segments. As depicted, a total of 16 path segments are presented in Fig. 9.

N: Set of intersection nodes in the AGV path map.

π_n: Set used to record path segments that terminate at intersection node $n, n \in N$. E.g., $\pi_2 = \{1, 10\}$ in Fig. 9.

ψ_i: Set that contains travel path segments for transporting container i from quayside to its designated block in the yard, $\psi_i \in P$. Note that the path segments in ψ_i are arranged in ascending order based on the sequence in which the AGV traverses them. $\psi_i(k)$ represents the kth path segment that the AGV navigates while transporting container i from the quayside to its designated block in the yard ($k = 1, 2, \ldots, |\psi_i|$).

σ_{ij}: Set that contains travel path segments of the AGV from the designated yard block of container i to the unloading position of container j at the quayside. Note that the path segments in σ_{ij} are arranged in ascending order based on the sequence in which the AGV traverses them. $\sigma_{ij}(k)$ represents the kth path segments that the AGV navigates while traveling from the designated block of container i to the unloading position of container j at the quayside ($k = 1, 2, \ldots, |\sigma_{ij}|$).

D: Set of discrete time units within the time horizon, $D = \{1, 2, \ldots, |D|\}$, where $|D|$ represents the time horizon for the whole unloading process.

We assume that the AGV always chooses the shortest route for transportation between any two points of the quayside and the yard, thus, we can obtain the set ψ_i for each container i, $i \in \Omega$, as well as the set σ_{ij} for each container pair (i, j), $i, j \in \Omega$.

Parameters

V: number of available AGVs.

C: number of transfer buffers/AGV-mates at the seaside end of a yard block.

K_p: the maximum capacity of path segment p, $p \in P$.

s_i: TQC processing time for executing a single-lift for container i, $i \in \Omega$.

s_i^{tan}: TQC processing time for executing a tandem-lift for container i, $i \in \Omega$.

e_i: ASC processing time for container i, $i \in \Omega$.

d_i: extra ASC processing time caused by ASC interactions within a block for container i, $i \in \Omega$. The values are calculated through formula (45) in Appendix.

u: AGV processing time under the AGV-mate.

$\tau(p)$: AGV travel time required to traverse path segment p, $p \in P$.

a_{ib}: index to record the designated yard blocks of containers: if container i is allocated to block b, $a_{ib} = 1$; otherwise, $a_{ib} = 0$.

β_{ip}^{loaded}: index to record whether an AGV route from the quayside to the yard involves a specific path segment: if the AGV transporting container i from the quayside to the yard is required to traverse path segment p, $\beta_{ip}^{loaded} = 1$; otherwise, $\beta_{ip}^{loaded} = 0$.

β_{ijp}^{empty}: index to record whether an AGV route from the yard to the quayside involves a specific path segment: if the AGV traveling from the yard block of container i to the unloading position of container j at the quayside is required to traverse path segment p, $\beta_{ijp}^{empty} = 1$; otherwise, $\beta_{ijp}^{empty} = 0$.

O: dummy starting task of AGVs.

$O\prime$: dummy ending task of AGVs.

M: a sufficiently large positive integer.

Decision variables

$x_{ij} \in \{0, 1\}$: where $x_{ij} = 1$ if an AGV, which just handles container i is scheduled to handle container j.

$z_{ikb} \in \{0, 1\}$: where $z_{ikb} = 1$ if container i is the kth container that arrives at the transfer buffers/AGV-mates at the seaside end of yard block i.

$y_i \in \{0, 1\}$: where $y_i = 1$ if the TQC executes a tandem-lift for container i, i.e., unloads containers i and $i + 1$ simultaneously.

q_i: the time when the TQC starts unloading container i; $t_i \geq 0$.

T_i: the time when the AGV-mate starts processing container i; $T_i \geq 0$.

m_i: the time when an ASC discharges container i from the AGV-mate; $m_i \geq 0$.

h_{ip}^{start}: the time at which the AGV transporting container i from the quayside to the yard starts traversing path segment p. If the route does not involve path segment p, then $h_{ip}^{start} = 0$.

h_{ip}^{end}: the time at which the AGV transporting container i from the quayside to the yard leaves path segment p. If the route does not involve path segment p, then $h_{ip}^{end} = 0$. h_{ip}^{end} also represents the time when the AGV arrives at the corresponding intersection node.

r_{ip}^{start}: the time at which the AGV (which just completed the transport task of container i) traveling from the yard block to the quayside starts traversing path segment p, if such route does not involve path segment p, then $r_{ip}^{start} = 0$.

r_{ip}^{end}: the time at which the AGV (which just completed the transport task of container i) traveling from the yard block to the quayside leaves path segment p, if such route does not involve path segment p, then $r_{ip}^{end} = 0$. r_{ip}^{end} also represents the time when the AGV arrives at the corresponding intersection node.

$w_{ipt}^{loaded} \in \{0,1\}$: where $w_{ipt}^{loaded} = 1$ if the AGV transporting container i from the quayside to the yard block is on path segment p at time t.

$w_{ipt}^{empty} \in \{0,1\}$: where $w_{ipt}^{empt} = 1$ if the AGV (which just completed the transport task of container i) traveling from the yard block to the quayside is on path segment p at time t.

$l_{ipt}^{loaded} \in \{0,1\}$: where $l_{ipt}^{loaded} = 1$ if the AGV transporting container i from the quayside to the yard leaves path segment p at time t.

$l_{ipt}^{empty} \in \{0,1\}$: where $l_{ipt}^{empty} = 1$ if the AGV (which just completed the transport task of container i) traveling from the yard block to the quayside leaves path segment p at time t.

F: the competition time of unloading operations by the TQCs; $F \geq 0$.

With the above descriptions and notations, the AGV scheduling problem for serving TQCs can be formulated as follows:

$$Minimize\ F \tag{1}$$

$$F \geq q_i + s_i - y_{i-1} \times M, \forall\ i \in \Phi \tag{2}$$

$$F \geq q_i + s_{i-1}^{tan} - (1 - y_{i-1})M, \forall\ i \in \Phi, i-1 \in \theta \tag{3}$$

$$\sum_{j \in \Omega/i \cup O\prime} x_{ij} = 1, \forall\ i \in \Omega \tag{4}$$

$$\sum_{j \in \Omega/i \cup O} x_{ji} = 1, \forall\ i \in \Omega \tag{5}$$

$$\sum_{i \in \Omega} x_{Oi} \leq V \tag{6}$$

$$y_i = 0, \forall\ i \in \Omega, i \notin \theta \tag{7}$$

$$y_i + y_{i+1} \leq 1, \forall\ i \in \theta \tag{8}$$

$$q_{i+1} - (1 - y_i)M \leq q_i + s_i(1 - y_i) \leq q_{i+1},\ \forall\ i \in \Omega, i \notin \Phi \tag{9}$$

$$h_{i\psi_i(1)}^{start} \geq q_i + s_i - (y_i + y_{i-1})M,\ \forall i \in \Omega \tag{10}$$

$$h_{i\psi_i(1)}^{start} \geq q_i + s_i^{tan} - (1 - y_i)M,\ \forall i \in \Omega \tag{11}$$

$$h_{i\psi_i(1)}^{start} \geq q_i + s_{i-1}^{tan} - (1 - y_{i-1})M,\ \forall i \in \Omega \tag{12}$$

$$h_{i\psi_i(k)}^{end} \geq h_{i\psi_i(k)}^{start} + \tau(\psi_i(k)),\ \forall i \in \Omega, \forall k = 1, 2, \ldots, |\psi_i| \tag{13}$$

$$h_{i\psi_i(k)}^{start} = h_{i\psi_i(k-1)}^{end} + 1,\ \forall i \in \Omega, \forall k = 2, \ldots, |\psi_i| \tag{14}$$

$$T_i \geq h_{i\psi_i(|\psi_i|)}^{end},\ \forall i \in \Omega \tag{15}$$

$$r_{i\sigma_{ij}(1)}^{start} \geq T_i + u - (1 - x_{ij})M,\ \forall i \in \Omega, \forall j \in \Omega \tag{16}$$

$$r_{i\sigma_{ij}(k)}^{end} \geq r_{i\sigma_{ij}(k)}^{start} + \tau(\sigma_{ij}(k)) - (1 - x_{ij})M,\ \forall i \in \Omega, \forall j \in \Omega, \forall k = 1, 2, \ldots, |\sigma_{ij}| \tag{17}$$

$$r_{i\sigma_{ij}(k-1)}^{end} - (1 - x_{ij})M \leq r_{i\sigma_{ij}(k)}^{start} - 1 \leq r_{i\sigma_{ij}(k-1)}^{end} + (1 - x_{ij})M, \quad \forall i \in \Omega, j \in \Omega, \forall k = 2, \ldots |\sigma_{ij}| \tag{18}$$

$$q_j + s_j \geq r_{i\sigma_{ij}(|\sigma_{ij}|)}^{end} - (1 - x_{ij})M - (y_j + y_{j-1})M,\ \forall i \in \Omega, \forall j \in \Omega \tag{19}$$

$$q_j + s_j^{tan} \geq r_{i\sigma_{ij}(|\sigma_{ij}|)}^{end} - (1 - x_{ij})M - (1 - y_j)M,\ \forall i \in \Omega, \forall j \in \Omega \tag{20}$$

$$q_j + s_{j-1}^{tan} \geq r_{i\sigma_{ij}(|\sigma_{ij}|)}^{end} - (1 - x_{ij})M - (1 - y_{j-1})M,\ \forall i \in \Omega, \forall j \in \Omega \tag{21}$$

$$w_{ipt}^{loaded} \geq \frac{1}{M^2}(t - h_{ip}^{start} + 1)(h_{ip}^{end} - t),\ \forall i \in \Omega, \forall p \in P, \forall t \in D \tag{22}$$

$$w_{ipt}^{empty} \geq \frac{1}{M^2}(t - r_{ip}^{start} + 1)(r_{ip}^{end} - t),\ \forall i \in \Omega, \forall p \in P, \forall t \in D \tag{23}$$

$$\sum_{i \in \Omega} w_{ipt}^{loaded} + \sum_{i \in \Omega} w_{ipt}^{empty} \leq K_p,\ \forall p \in P, \forall t \in D \tag{24}$$

$$\sum_{t \in D} t l_{ipt}^{loaded} = h_{ip}^{end},\ \forall\ i \in \Omega, \forall p \in P \tag{25}$$

$$\sum_{t \in D} t l_{ipt}^{empty} = r_{ip}^{end},\ \forall\ i \in \Omega, \forall p \in P \tag{26}$$

$$\sum_{t \in D} l_{ipt}^{loaded} \leq 1,\ \forall\ i \in \Omega, \forall p \in P \tag{27}$$

$$\sum_{t \in D} l_{ipt}^{empt} \leq 1,\ \forall\ i \in \Omega, \forall p \in P \tag{28}$$

$$\sum_{i \in \Omega} \sum_{p \in \pi_n} l_{ipt}^{empty} + \sum_{i \in \Omega} \sum_{p \in \pi_n} l_{ipt}^{loaded} \leq 1,\ \forall n \in N, \forall t \in D \tag{29}$$

$$\sum_{k \in \Omega} z_{ikb} \geq a_{ib}, \forall i \in \Omega,\ \forall\ b \in B \tag{30}$$

$$\sum_{i \in \Omega} z_{ikb} \leq 1, \forall k \in \Omega,\ \forall\ b \in B \tag{31}$$

$$\sum_{i \in \Omega} z_{ikb} \geq \sum_{i \in \Omega} z_{i(k+1)b}, \forall k \in \Omega, \forall b \in B \tag{32}$$

$$T_i \leq T_j + (2 - z_{ikb} - z_{j(k+1)b})M, \forall i \in \Omega, \forall j \in \Omega, \forall k \in \Omega, \forall b \in B \tag{33}$$

$$m_i \geq T_i + u, \forall i \in \Omega \tag{34}$$

$$m_i + e_i + d_i \leq m_j + \left(2 - z_{ikb} - z_{j(k+1)b}\right) M, \forall i \in \Omega, \forall j \in \Omega, \quad \forall k \in \Omega, \forall b \in B \tag{35}$$

$$m_i \leq T_j + \left(2 - z_{ikb} - z_{j(k+C)b}\right) M, \forall i \in \Omega, \forall j \in \Omega, \forall k \in \Omega, \forall b \in B \tag{36}$$

$$r_{ip}^{start} \leq r_{ip}^{end} \leq \sum_{j \in \Omega} x_{ij} \times \beta_{ijp}^{empty} \times M,\ \forall i \in \Omega, \forall p \in P \tag{37}$$

$$h_{ip}^{start} \leq h_{ip}^{end} \leq \beta_{ip}^{loaded} \times M,\ \forall i \in \Omega, \forall p \in P \tag{38}$$

The objective function (1) minimizes the maximum completion time (makespan) of TQCs. Constraints (2) and (3) define the makespan of TQCs, guaranteeing that the makespan is greater than or equal to the completion time for unloading the final task of each TQC. Constraint (2) is invoked when the last task of a specific TQC is unloaded by a single-lift, while constraint (3) is invoked when the last task of a given TQC is unloaded in a tandem-lift (i.e., container i is unloaded simultaneously with container $i-1$). Constraints (4) and (5) present that each container in Ω has a predecessor and a successor task on the same AGV, which can ensure that each container is transported once and the AGV travel balance. Constraint (6) ensures that the total number of AGVs to be used does not exceed the maximum number of available AGVs. Constraints (7) and (8) regulate the execution of tandem-lifts: constraint (7) ensures that only containers permitting tandem-lift operations can be subjected to tandem-lifts, while constraint (8) restricts the feasibility of tandem-lifts for neighboring containers. As illustrated in Fig. 8, the TQC is unable to perform tandem-lifts for both containers 1 and 2. Constraint (9) ensures the operational sequence and continuity of TQCs: if the TQC executes a tandem-lift for container i, then containers i and $i+1$ are unloaded simultaneously, i.e., $q_i = q_{i+1}$; otherwise,

the unloading time of container $i+1$ is greater than or equal to the unloading time of container i plus the processing time of container i by the TQC. Constraints (10)–(15) ensure the time consistency of AGVs for transporting containers from the quayside to the yard: constraints (10)–(12) stipulate that the start time of AGV transportation (from the quayside) must be greater than or equal to the time at which the container is unloaded on the AGV. Constraint (10) works when container i is unloaded in a single-lift, constraint (11) works when container i is unloaded in a tandem-lift with container $i+1$ (i.e., $y_i = 1$), and constraint (12) is invoked when container i is unloaded in a tandem-lift with container $i-1$ (i.e., $y_{i-1} = 1$). Constraints (13) and (14) define the continuity of AGV transportation between different path segments, where '+1' in constraint (14) represents a one-time unit spent on the intersection node. Constraint (15) signifies that the start time of the AGV-mate operation should be greater than or equal to the time the AGV arrives at the designated yard block. Constraints (16)–(21) enforce the continuity of AGV transportation from the yard to the quayside: constraint (16) stipulates that the start time of AGV transportation (from the yard) must be greater than or equal to the completion time at which the previous container on the AGV was processed by the AGV-mate. Constraints (17) and (18) define the continuity of AGV transportation between different path segments from the yard to the quayside. Constraints (19)–(21) signifies the coordination between AGVs and TQCs, which ensures that only the AGV arrives at the TQC can the TQC unload the container on the AGV. Constraint (19) works when container i is unloaded in a single-lift, constraint (20) works when container i is unloaded in a tandem-lift with container $i+1$ (i.e., $y_i = 1$), and constraint (21) is invoked when container i is unloaded in a tandem-lift with container $i-1$ (i.e., $y_{i-1} = 1$). Constraints (22) and (23) define variables w_{ipt}^{loaded} and w_{ipt}^{empty}: if $t \in [h_{ip}^{start}, h_{ip}^{end})$, then $w_{ipt}^{loaded} = 1$; if $t \in [r_{ip}^{start}, r_{ip}^{end})$, then $w_{ipt}^{empty} = 1$. Constraint (24) ensures the path capacity restrictions. Constraints (25)–(28) confine and calculate variables l_{ipt}^{loaded} and l_{ipt}^{empty}. Constraint (29) mitigates node conflicts by restricting that at most one AGV can traverse an intersection node at the same time. Constraint (30) builds up relations between variable z_{ikb} and parameter a_{ib}. Constraint (31) ensures that at most one container can arrive at a block with a certain order. Constraint (32) ensures the continuity of the arrived sequence of containers at a block. Constraints (33) confines relations between variables z_{ikb} and T_i: if container i is the kth container that arrives at block b and container j is the $(k+1)$-th container that arrives at block b, then, T_j should be greater than or equal to T_i. Constraint (34) builds up relations between variables m_i and T_i. Constraint (35) implies that an ASC processes only one container at a time and the processing sequence is based on the arrived order of containers at the AGV-mates. Constraint (36) ensures the limited capacity of transfer buffers: when all transfer buffers are busy at the seaside end of a block, the arrived AGV at that block should wait until a transfer buffer is available. Specifically, if container i is the kth task that arrives at block b and container j is the $(k+C)$-th task that arrives at block b, then the execution time of container j on the AGV-mate should be greater than or equal to the retrieval time of container i by the ASC. Constraints (37) and (38) ensure that if a route does not traverse a specific path segment, then both the starting and ending times for the corresponding AGV using that path segment are set to zero.

Note that constraint (35) restricts the ASC to processing containers that arrived at transfer buffers according to the arrival order of the containers, this is equivalent to the situation where the ASC processes containers that arrived at transfer buffers with a random order. This is because all AGVs are identical, and AGVs arriving at the same yard block share an identical starting point before proceeding to execute subsequent tasks. Under this premise, the capacity of transfer buffers can be well confined by constraint (36).

We provide an example to assist readers in gaining a clearer comprehension of the correlations between different time parameters and variables, as well as the interactions among the TQC, AGVs, and AGV-mates that are reflected by the constraints. In Fig. 10, the TQC executes a tandem-lift for containers j and $j+1$; container j is assigned to AGV1, and the previous task on AGV1 is container i; container $j+1$ is assigned to AGV2, and the previous task on AGV2 is container i'. As seen in Fig. 10, AGV1 (AGV2) travel begins from the yard block at time $h_{i\sigma_{ij}(1)}^{start}$ ($h_{i'\sigma_{i'j+1}(1)}^{start}$), which should be greater than or equal to the completion time of its last task, i.e., $T_i + u$ ($T_{i'} + u$). The orange dashed lines illustrate the interaction relationship between AGVs and AGV-mates, which are reflected by constraint (16). Then AGV1 (AGV2) moves towards the designated position of container j ($j+1$) at the quay. The moment at which AGV1 (AGV2) arrives at the quay is time $h_{i\sigma_{ij}(|\sigma_{ij}|)}^{end}$ ($h_{i'\sigma_{i'j+1}(|\sigma_{i'j+1}|)}^{end}$), which is confined by constraints (17) and (18). Because containers j and $j+1$ are unloaded simultaneously in a tandem-lift, we have $q_j = q_{j+1}$, which is ensured by constraint (9). And the containers can only be unloaded onto the AGVs when both AGVs have arrived, i.e., $q_j + s_j^{tan} \geq h_{i\sigma_{ij}(|\sigma_{ij}|)}^{start}$, $q_{j+1} + s_j^{tan} \geq h_{i'\sigma_{i'j+1}(|\sigma_{i'j+1}|)}^{end}$, which are reflected by constraints (19)–(21). Subsequently, the two AGVs depart from the quay and transport the containers to the yard. They depart at time $h_{j\psi_j(1)}^{start}$, $h_{j+1\psi_{j+1}(1)}^{start}$, respectively, which should be greater than or equal to $q_j + s_j^{tan}$ (reflected by constraints (10)–(12)). The green dashed line illustrates the interaction relationships between the AGV and TQC. The arrival time of AGV1 (AGV2) at the yard is $h_{j\psi_j(|\psi_j|)}^{end}$ ($h_{j+1\psi_{j+1}(|\psi_{j+1}|)}^{end}$), which is confined by constraints (13)–(14). Only after the AGV arrives at the yard, can AGV-mate starts processing the container on the AGV, i.e, T_j should be greater than or equal to $h_{j\psi_j(|\psi_j|)}^{end}$. The red dashed lines illustrate the relationships between AGV and AGV-mates, which is ensured by constraint (15).

Due to the existence of constraints (22) and (23), the above-mentioned model is nonlinear. Consequently, our next step involves linearizing the two constraints, thereby transforming the model into a mixed-integer linear programming model (MILP), which can be solved using programming solvers.

Two more 0–1 variables are introduced as follows:

$b_{ipt}^{loaded} \in \{0,1\}$: where $b_{ipt}^{loaded} = 1$ if the AGV transporting container i from the quayside to the yard begins to pass through path segment p at time t.

$b_{ip}^{empty} \in \{0,1\}$: where $b_{ipt}^{empty} = 1$ if the AGV (which just completed the transport task of container i) traveling from the yard block to the quayside begins to pass through path segment p at time t.

Then, constraints (22) and (23) will be replaced by the following set of constraints.

$$\sum_{t\in D} t b_{ipt}^{loaded} = h_{ip}^{start},\ \forall\, i \in \Omega, \forall p \in P \tag{39}$$

$$\sum_{t\in D} t b_{ipt}^{empty} = r_{ip}^{start},\ \forall\, i \in \Omega, \forall p \in P \tag{40}$$

$$\sum_{t\in D} b_{ipt}^{loaded} \leq 1,\ \forall\, i \in \Omega, \forall p \in P \tag{41}$$

$$\sum_{t\in D} b_{ipt}^{empty} \leq 1,\ \forall\, i \in \Omega, \forall p \in P \tag{42}$$

$$w_{ipt}^{loaded} \geq \sum_{t'=t+1}^{|D|} l_{ipt'}^{loaded} - \sum_{t'=t+1}^{|D|} b_{ipt'}^{loaded},\ \forall i \in \Omega, \forall p \in P, \forall t \in D \tag{43}$$

$$w_{ipt}^{empty} \geq \sum_{t'=t+1}^{|D|} l_{ipt'}^{empty} - \sum_{t'=t+1}^{|D|} b_{ipt'}^{empty},\ \forall i \in \Omega, \forall p \in P, \forall t \in D \tag{44}$$

Constraints (39)–(42) confine and calculate variables b_{ipt}^{loaded} and b_{ipt}^{empty}. By utilizing variables l_{ipt}^{loaded}, l_{ipt}^{empty}, b_{ipt}^{loaded} and b_{ipt}^{empty}, Constraints (43) and (44) can represent the same meaning as constraints (22) and (23), i.e., if $t \in [h_{ip}^{start}, h_{ip}^{end})$, then $w_{ipt}^{loaded} = 1$; if $t \in [r_{ip}^{start}, r_{ip}^{end})$, then $w_{ipt}^{empty} = 1$.

Through the aforementioned transformation, we have established a MILP model that can be solved by the programming solver. The 'big-M' constraints (which refer to the constraints that contain parameter M) in the model (e.g., constraints (9), (10), etc.) may greatly impact the performance of the programming solver, and the smallest feasible

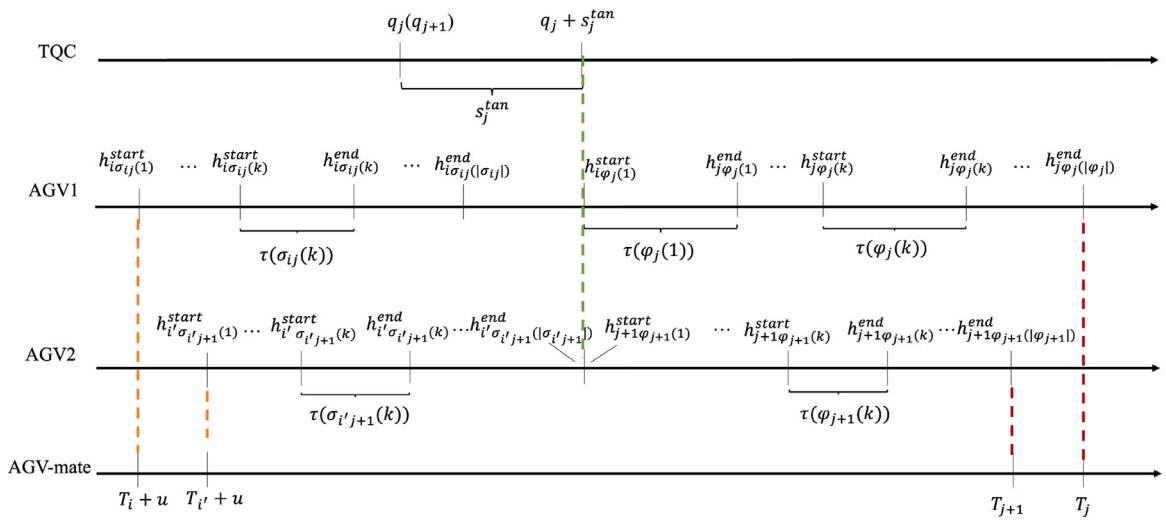


Fig. 10. An illustrative example of correlations between different time parameters and variables.

value should be set for M. The smallest feasible value for the M in all 'big-M' constraints can be regarded as the minimum completion time of the whole unloading process (i.e., the time when the last container is unloaded by the ASC in the yard). However, because of the interconnections among multiple AGVs, TQCs, AGV-mates, and ASCs, combined with the congestion and conflicts among AGVs, it becomes impractical to directly express the minimum completion time using mathematical formulas. Therefore, we obtain a feasible solution to the model using heuristic rules (i.e., the *Construction algorithm* presented in Section 4.2) and calculate the completion time of the schedule related to that feasible solution, utilizing it as the value for parameter M in the model. Besides, the value of parameter $|D|$, i.e., the time horizon for the whole unloading process, is also set to the same value as M for reducing the number of variables and constraints in the MILP model.

In the unloading operations at ACTs, many AGVs are involved in executing a large number of container transporting tasks. Because the proposed MILP model is only solvable for small-scale instances (see Section 5.2), we develop a heuristic algorithm for solving realistic-scale instances efficiently in Section 4.

4. Solution algorithm

This section presents a heuristic algorithm based on the variable neighborhood search (VNS) to solve the AGV scheduling problem for serving the TQCs. The VNS is a heuristic method proposed by Mladenovi and Hansen (1997) to solve large-scale combinatorial optimization problems, which exploits the idea of systematically changing the neighborhood structures during the search to extensively explore the search space. The literature shows the VNS can obtain good performance regarding vehicle routing problems (such as Kytoejoki et al., 2007; Stenger et al., 2014; Christof and Kenneth, 2017; Sadati and Atay, 2021; Bezerra et al., 2022). AGV scheduling problem can be seen as a special vehicle routing problem arising at ACTs with complicated constraints. Therefore, in this study, we develop a heuristic algorithm based on the idea of VNS with specific local search procedures related to the problem specificities.

4.1. Algorithm framework

An overview of the heuristic algorithm is provided in Table 1. The algorithm can be regarded as a multi-start local search algorithm based on the concept of VNS, referred to as the MS-LS algorithm. The MS-LS algorithm begins by constructing an initial solution $S_{initial}$ using the *Construction algorithm* introduced in Section 4.2, and $S_{initial}$ is recorded as the best global solution S_{global} obtained so far. Then, S_{global} is updated through iterations with a number of $maxIter$ (steps 4 to 24 in Table 1). During each iteration, a new initial solution is generated and recorded as the best solution in the current iteration, S_{best}. The initial solution in a certain iteration is generated by destroying the part of the scheme of S_{global} randomly to obtain a partial solution $S_{partial}$ (see *Destroy algorithm* in Section 4.3) and repairing $S_{partial}$ using the *Construction algorithm*. Then S_{best} is updated using the *Local search algorithm* presented in Section 4.4.

The *Local search algorithm* endeavors to enhance the initial solution of each iteration by employing the variable neighborhood descent (VND) procedure of the VNS. In the VND, three neighborhood structures are designed: NS_1, NS_2, and NS_3. The local search is initiated using the NS_k neighborhood structure, with k set as 1. Through exploration of the $N_k(S_{best})$ neighborhood, the optimal solution within the current neighborhood, labeled as S_{new}, is derived and recorded. Subsequently, S_{new} is improved using the function $Seq\_Exchange()$ (introduced in Section 4.4). If S_{new} yields an improvement over S_{best} (i.e., $F(S_{new}) < F(S_{best})$), S_{best} is updated, and k is reset to 1. Otherwise, k is incremented ($k \leftarrow k+1$), indicating a shift to the next neighborhood structure. The execution of the VND procedure persists until none of the three neighborhood structures produces an enhanced solution (i.e., $k = 4$). If S_{best} proves superior to the current best global solution obtained (i.e., $F(S_{best}) < F(S_{global})$), then S_{global} is updated accordingly.

4.2. Construction algorithm

The *Construction algorithm* builds up a feasible solution to the problem. To accurately capture the unique characteristics of the problem, we have adopted a three-layer encoding strategy to represent the configuration of a solution. In the first layer, each value corresponds to the index of a container. The second layer contains the index of the AGV, indicating the assignment of AGVs to individual containers. The sequencing of AGV operations for the containers adheres to the order in which the containers are listed in the first layer. The third layer reflects the tandem-lift operation of the TQCs: a value of 1 indicates that the corresponding container is executed with a tandem-lift by the TQC.

To provide clarity, consider the scenario depicted in Fig. 11, which encompasses 10 containers and 3 AGVs. The left diagram illustrates an encoding structure, which can be translated into the AGV schedule displayed in the corresponding right diagram. Note that in the right

Table 1
The framework of MS-LS algorithm.

Obtain an initial solution $S_{initial}$ using the *Construction algorithm* (see Section 4.2)
Record the best global solution obtained so far $S_{global} \leftarrow S_{initial}$
Initiate the iteration index $Iter \leftarrow 0$
While $Iter < maxIter$ **do**
  Delete part of scheme in S_{global} randomly using the *Destroy algorithm* (see Section 4.3) to obtain a partial solution $S_{partial}$
  Repair $S_{partial}$ using the *Construction algorithm* to obtain a feasible solution, which is regarded as the best solution obtained in the current iteration S_{best}
  Update S_{best} using the *Local Search algorithm* (see Section 4.4):
  $k \leftarrow 1$
  While $k \leq 3$ **do**
    Search for the best solution in $NS_k(S_{best})$, record it as S_{new}
    Further improve S_{new} by $S_{new} \leftarrow Seq\_Exchange(S_{new})$
    If $F(S_{new}) < F(S_{best})$ **then**
      $S_{best} \leftarrow S_{new}$
      $k \leftarrow 1$
    Else
      $k \leftarrow k + 1$
    End if
  End while
  Update the best global solution:
  If $F(S_{best}) < F(S_{global})$ **then**
    $S_{global} \leftarrow S_{best}$
  End if
  $Iter \leftarrow Iter + 1$
End while

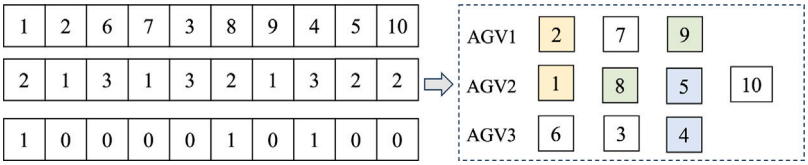


Fig. 11. The encoding structure of a solution and its corresponding real-world significance.

diagram of Fig. 11, containers that share the same color are indicative of two containers being unloaded in a tandem-lift simultaneously.

Given the fixed operational sequence of TQCs and the restrictions related to the tandem-lifts, the generation of a random three-layer encoding could potentially yield either an infeasible outcome or one of poor quality. Thus, we have designed the following algorithm to generate a feasible solution with high quality, while concurrently deriving the corresponding three-layer encoding.

The supplementary notations for the *Construction algorithm* are introduced as follows.

$readyTime_q$: ready time of the next task of TQC q;

$arrTime_v$: the time when AGV v arrives at the position of the current unloading task at the quayside.

$idleTime_v$: the time when AGV v completes its previous task (the time when the previous task of AGV v is unloaded by the AGV-mate).

$lastCon_v$: the previous task of AGV v, with an initial value of -1.

$unloadTime_i$: the time when container i is unloaded on the assigned AGV by the TQC.

$yardTime_i$: the time when container i arrives at its assigned yard block.

$mateTime_i$: the time when the AGV-mate unloads container i from the AGV.

$exeTime$: the time when an ASC starts to perform an unloading operation at AGV-mates.

$ASCTime_i$: the time when the ASC in the allocated block of container i completes its previous task (idle time of the ASC).

$PathOccupy_{pt}$: record the number of AGVs on path segment p at time t, with an initial value of 0.

$NodeOccupy_{nt}$: indicates the occupancy status of node n at time t. Assign a value of 1 if occupied; otherwise, assign a value of 0. The initial value is set to 0.

$Node_p$: the intersection node corresponding to the endpoint of path segment p.

Tan_i: the index to record whether a tandem-lift is executed for container i.

$startTime_{ip}$: the time at which the AGV traveling from the yard to the quayside for retrieving container i starts traversing path segment p.

$endTime_{ip}$: the time at which the AGV traveling from the yard to the quayside for retrieving container i leaves path segment p.

For the *Construction algorithm*, the solution is obtained by sequentially allocating unloading containers to the AGVs. The process of generating an initial feasible solution involves the following seven steps:

Step 1: Select the next container.

Obtain the ready time of the next container of each TQC ($readyTime_q, q \in Q$). Select a TQC greedily using the paradigm of roulette wheels according to the values of $readyTime_q$, and record the selected TQC as $selectQ$. The smaller the ready time of the next container, the more likely the related TQC is selected. Record the next container of $selectQ$ as con.

Step 2: Select the AGV.

Calculate $arrTime_v$, $v \in V$ by $arrTime_v = Cal\_arrTime(v, con)$, where $Cal\_arrTime(v, i)$ is a function that is used to calculate the arrival time of AGV v at the quayside for retrieving container i, taking into account congestion and conflicts among AGVs. The pseudocode of function $Cal\_arrTime(v, i)$ is provided in Table 2.

Select the AGV for serving container con using the paradigm of roulette wheels according to the values of $arrTime_v$. Record the selected AGV as $AGV1$.

Execute the transportation of $AGV1$ from the yard to the quayside to retrieve container con, and subsequently update $OccupyPath$ and $OccupyNode$ using the function $Update\_Occupy(AGV1, con)$. The pseudocode of function $Update\_Occupy(v, i)$ is presented in Table 3.

Step 3: Determine whether to execute a tandem-lift.

If $\theta_{con} = 0$, proceed to step 4; otherwise, record the container that may be unloaded with container con in a tandem-lift as $con2$

Table 2
The pseudocode of function $Cal\_arrTime(v,i)$.

```
If lastCon_v ≠ −1 then
    j ← lastCon_v                    \\obtain the previous task
    For k = 1 to |σ_ji| do
        p ← σ_ji(k)                  \\obtain the current path segment
        If k = 1 then
            startTime_ip ← idleTime_v
        Else
            p' ← σ_ji(k − 1)         \\obtain the previous path segment
            startTime_ip ← endTime_ip' + 1
        End if
        feaTime ← 0 \\record the minimum feasible time for entering the current path segment
        For t = |D| to 0 do          \\ensure path capacity constraints
            If PathOccupy_pt = K_p then
                feaTime ← t + 1
                break
            End if
        End for
        If startTime < feaTime:
            startTime ← feaTime
        End if
        If k > 1 then                \\ensure node conflict constraints
            endTime_ip' ← startTime_ip − 1
            While NodeOccupy_Node_p' endTime_ip' = 1 do
                startTime_ip ← startTime_ip + 1
                endTime_ip' ← startTime_ip − 1
            End while
        End if
        endTime_ip ← startTime_ip + τ(p)
    End for
Else
    endTime_ip ← idleTime_v
End if
return endTime_ip
```

Table 3
The pseudocode of function $Update\_Occupy(v,i)$.

```
If lastCon_v ≠ −1 then
    j ← lastCon_v                    \\obtain the previous task
    For k = 1 to |σ_ji| do
        p ← σ_ji(k)                  \\obtain the current path segment
        If k = 1 then
            startTime_ip ← idleTime_v
        Else
            p' ← σ_ji(k − 1)         \\obtain the previous path segment
            startTime_ip ← endTime_ip' + 1
        End if
        feaTime ← 0
        For t = |D| to 0 then
            If PathOccupy_pt = K_p then
                feaTime ← t + 1
                break
            End if
        End for
        If startTime < feaTime then
            startTime ← feaTime
        End if
        If k > 1 then
            temp ← endTime_ip'       \\record the original value of endTime_ip'
            endTime_ip' ← startTime_ip − 1
            While NodeOccupy_Node_p' endTime_ip' = 1 do
                startTime_ip ← startTime_ip + 1
                endTime_ip' ← startTime_ip − 1
            End while
            NodeOccupy_Node_p' endTime_ip' ← 1        \\update NodeOccupy
            For t = temp to endTime_ip' − 1 do        \\update PathOccupy
                PathOccupy_p't ← PathOccupy_p't + 1
            End for
        End if
        endTime_ip ← startTime_ip + τ(p)
        For t = startTime_ip to endTime_ip − 1 do     \\update PathOccupy
                PathOccupy_pt ← PathOccupy_pt + 1
        End for
    End for
End if
```

Table 4
The pseudocode of function $Cal\_Tan()$.

```
Input arrTime_AGV1, arrTime_AGV2, readyTime^tan_selectQ
If arrTime_AGV1 ≤ readyTime^tan_selectQ and arrTime_AGV2 ≤ readyTime^tan_selectQ then
    return 1
Else
    Generate a decimal number between 0 and 1 randomly, and record it as rr
    If (arrTime_AGV2 − max{arrTime_AGV1, readyTime_selectQ}) / s_con > rr then
        return 0
    Else
        return 1
    End if
End if
```

($con2 = con + 1$). Calculate $arrTime_v$ for container $con2$ ($v \in V, v \neq AGV1$) by $arrTime_v = Cal\_arrTime(v, con2)$. Next, apply the roulette wheel selection approach to greedily determine the AGV responsible for handling $con2$, based on the values of $arrTime_v$. Record the selected AGV as $AGV2$. Note that the two AGVs for serving the containers in a tandem-lift of the TQC should be different.

Calculate the actual ready time of $selectQ$ if $selectQ$ executes a tandem-lift for con by $readyTime^{tan}_{selectQ} = readyTime_{selectQ} - s_{con} + s^{tan}_{con}$. Calculate the value of Tan_{con} using the function $Cal\_tan()$. Refer to Table 4 for the pseudocode of the function $Cal\_tan()$. If $Tan_{con} = 0$, proceed to step 4; otherwise, move to step 5.

Step 4: Calculate the unloading time of container con on the AGV.

Append con to the first layer of the encoding, append $AGV1$ to the second layer of the encoding, and append '0' to the third layer of the encoding.

The unloading time of con on the AGV by the TQC should be:

$$unloadTime_{con} = \max\{readyTime_{selectQ}, arrTime_{AGV1}\}.$$

If there is a subsequent task for $selectQ$, update and store it as $nextT$, then adjust its associated ready time by

$$readyTime_{selectQ} = unloadTime_{con} + s_{nextT}.$$

Step 5: Calculate the unloading time of containers con and $con2$ on the AGVs.

Execute the transportation of $AGV2$ from the yard to the quayside to retrieve container $con2$, and update $OccupyPath$ and $OccupyNode$ using function $Update\_Occupy(AGV2, con2)$. Append con and $con2$ to the first layer of the encoding, append $AGV1$ and $AGV2$ to the second layer of the encoding, and append '1' and '0' to the third layer of the encoding.

The unloading time of con and $con2$ should be:

$$unloadTime_{con} = unloadTime_{con2} = \max\{readyTime^{tan}_{selectQ},\ arrTime_{AGV1},\ arrTime_{AGV2}\}$$

If there is a subsequent task for $selectQ$, update and store it as $nextT$, then adjust its associated ready time by

$$readyTime_{selectQ} = unloadTime_{con} + s_{nextT}.$$

Step 6: Calculate the unloading time of the current task by the AGV-mates.

The arrival time of AGV_1 from the quayside to the designated yard block of con can be calculated by $yardTime_{con} = Cal\_Update\_Occupy(AGV1, con)$. $Cal\_Update\_Occupy(v, i)$ is a function that calculates the arrival time of AGV v at the yard block of container i by considering congestion and conflicts among AGVs. Furthermore, this function updates the status of $NodeOccupy$ and $PathOccupy$. The process of $Cal\_Update\_Occupy(v, i)$ is analogous to the mechanisms outlined in Tables 2 and 3. Therefore, detailed descriptions of the function $Cal\_Update\_Occupy(v, i)$ are not provided.

If an unoccupied transfer buffer is available at the seaside end of the designated yard block of container con, AGV_1 can be served directly by an AGV-mate, and the time when con is unloaded by the AGV-mate can be calculated by $mateTime_{con} = yardTime_{con} + u$.

Otherwise, the ASC in the designated yard block of con should perform an unloading operation for a container at the AGV-mates (the container in the transfer buffers with the minimum arrival order is selected, and is recorded as con_s).

The execution time of ASC is calculated by

$$exeTime = \max\{mateTime_{con_s},\ ASCTime_{con_s}\}.$$

The time when con is unloaded by the AGV-mate can be calculated by

$$mateTime_{con} = \max\{yardTime_{con},\ exeTime\} + u.$$

The idle time of the related ASC is updated by $ASCTime_{cons} = exeTime + e_{con_s} + d_{con_s}$.

The idle time of AGV_1 is updated by $idleTime_{AGV1} = mateTime_{con}$.

The process for calculating the completion time of unloading $con2$ (if it exists) at the AGV-mates follows the same procedures as those applied for con.

Step 7: Repeat Steps 1 to 6 or terminate with a feasible solution.

The above procedures (steps 1 to 6) are repeated until all the containers are unloaded by the TQCs; if all the containers are unloaded, a feasible solution is obtained and the objective is calculated by $F = \max_{i\in\Omega}\{unloadTime_i\}$.

In the *Construction algorithm*, a solution, as well as its corresponding encoding, can be obtained by following steps 1 through 7. This process ensures the feasibility of the solution and the quality of the initial solution. Furthermore, when dealing with a known encoding, the process of calculating the objective value adheres to a sequence analogous to the steps from 1 to 7. In this context, the processing order of containers, AGV assignments and operation sequences, and the operational mode of the TQC for each container are already established. Consequently, there is no requirement for random generation.

4.3. Destroy algorithm

For obtaining a diversified initial solution in each iteration, and maintaining the characteristics of the best global solution obtained so far, we employ a *Destroy algorithm* that transforms the best global solution into a partial solution. The procedures are as follows.

Randomly choose a container from the initial layer of the encoding of the best global solution. Subsequently, remove all containers that come after the chosen container, including the selected container itself. At the same time, erase the associated information from the second and third layers of the encoding. Note that if two containers are to be unloaded simultaneously in a tandem-lift, it is essential to ensure that either both containers are retained or both containers are removed.

After a partial solution is obtained, it is repaired by the *Construction algorithm* presented in Section 4.2 and then, an initial solution in a certain iteration of the MS-LS algorithm could be generated.

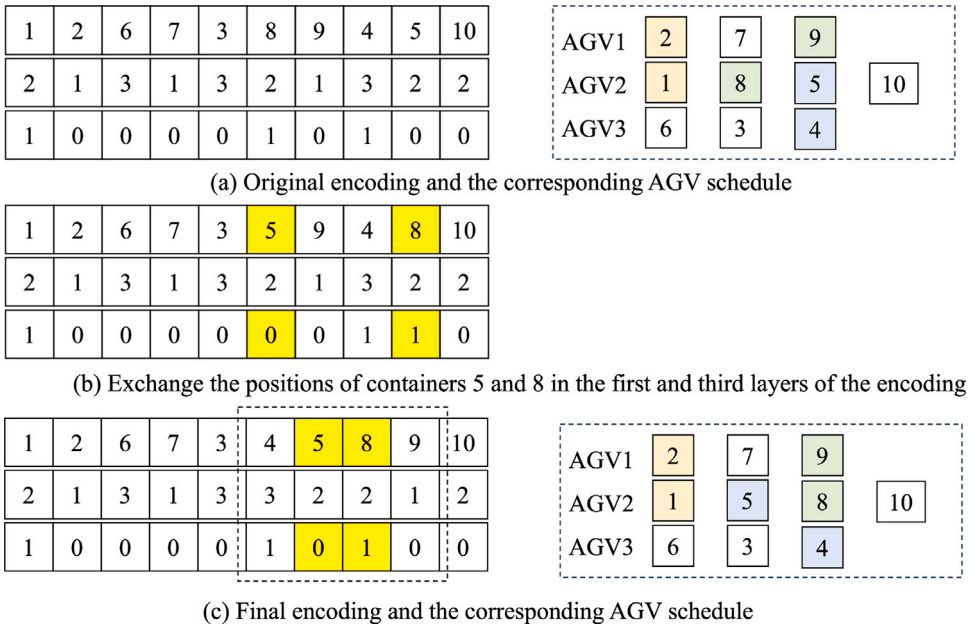


Fig. 12. An example of an operation in NS_1.

4.4. Local search algorithm

The *Local search algorithm* aims to improve the initial solution obtained in each iteration. This algorithm draws inspiration from the VND procedure of the VNS. Within the VND, three neighborhood structures have been devised, which are introduced as follows.

(1) NS_1: Exchange the positions of two containers in the first layer of encoding, and simultaneously swap the corresponding values in the third layer of encoding. This operation serves a dual purpose. First, it facilitates the swapping of two task assignments between two AGVs. Second, it allows for the rearrangement of task sequences within a single AGV. As a result, this operation encompasses a substantial search space. It is important to note that while performing the swapping process, the positions of the two containers being jointly unloaded in a tandem-lift should always remain adjacent.

An illustrative example is provided in Fig. 12 to clarify the above concept. The original encoding of a solution along with its corresponding AGV schedule is depicted in Fig. 12(a). Let us consider a scenario where a move in NS_1 involves swapping the positions of containers 5 and 8. This leads to a modified encoding as depicted in Fig. 12(b). However, an issue arises in the modified scenario shown in Fig. 12(b), where containers 4 and 5 (as well as containers 8 and 9) are being unloaded simultaneously in a tandem-lift. To ensure proper sequencing, adjustments need to be made to maintain adjacency among containers involved in the same tandem-lift. It is crucial to note that these adjustments must be executed without altering the AGV assignments and TQC operations for the containers. Consequently, the final adjusted encoding, along with its corresponding AGV schedule, is achieved, as demonstrated in Fig. 12(c).

After the exchange, a crucial step involves validating the feasibility of the new encoding. We validate the feasibility of the new encoding by checking the feasibility of its corresponding AGV schedule. The AGV schedule is considered feasible if it adheres to the following criteria: (1) it avoids containing two containers in the same tandem-lift operation in a route; (2) it adheres to the unloading sequence requirements of the TQCs.

If the new encoding is feasible, we can calculate its corresponding objective value similar to steps 1–7 in Section 4.2. In this case, the processing order of containers, AGV assignments and operation sequences, and the operational mode of the TQC for each container are all derived from the encoding itself, as opposed to being generated at random.

(2) NS_2: Change a value in the second layer of encoding, while keeping the first and third layers of encoding unchanged. An operation within NS_2 is equivalent to removing a container from one AGV and inserting it into the schedule of another AGV. Note that the value in the second layer should be no more than the total number of AGVs. Similar to the process in NS_1, after a move in NS_2 is executed, the feasibility of the new encoding should be checked and the objective value can be calculated if the new encoding is feasible.

(3) NS_3: Modify the values within the third layer of encoding. It has the capability to alter the operational mode of the TQC associated with the relevant container(s).

Take container i as an example, if its corresponding value in the third layer of encoding is '1' (which represents that containers i and i+1 should be unloaded simultaneously in a tandem-lift), we can change '1' to '0', then containers i and $i+1$ will be unloaded sequentially by single-lifts of the TQC.

If the corresponding value of container i in the third layer of encoding is '0', a subsequent step entails an assessment of whether it can be transformed into '1'. Specifically, if $\theta_i = 1$ (indicating the feasibility of such an alteration) and containers i and $i+1$ have been designated to distinct AGVs, the conversion from '0' to '1' can be performed. Then, it is crucial to ensure that the corresponding values for containers $i-1$ and $i+1$ in the third layer of encoding are modified to '0', to ensure the new encoding satisfies constraint (8).

An instance of an operation within NS_3 is depicted in Fig. 13. The initial encoding of a solution and its corresponding AGV schedule are shown in Fig. 13(a). In this case, let us modify the value within the third layer of encoding corresponding to container 7 to '1'. This transformation results in the encoding shown in Fig. 13(b). However, as exhibited in Fig. 13(b), containers 7 and 8, as well as containers 8 and 9, are subjected to simultaneous unloading via a tandem-lift, a situation that contradicts constraint (8). To address this issue, it is necessary to modify the value within the third layer of encoding corresponding to container 8 to '0', as demonstrated in Fig. 13(c). Then, adjustments should be made to keep the containers in a tandem-lift adjacent (the AGV assignments and TQC operations for containers should be maintained unchanged during the adjustment process), the

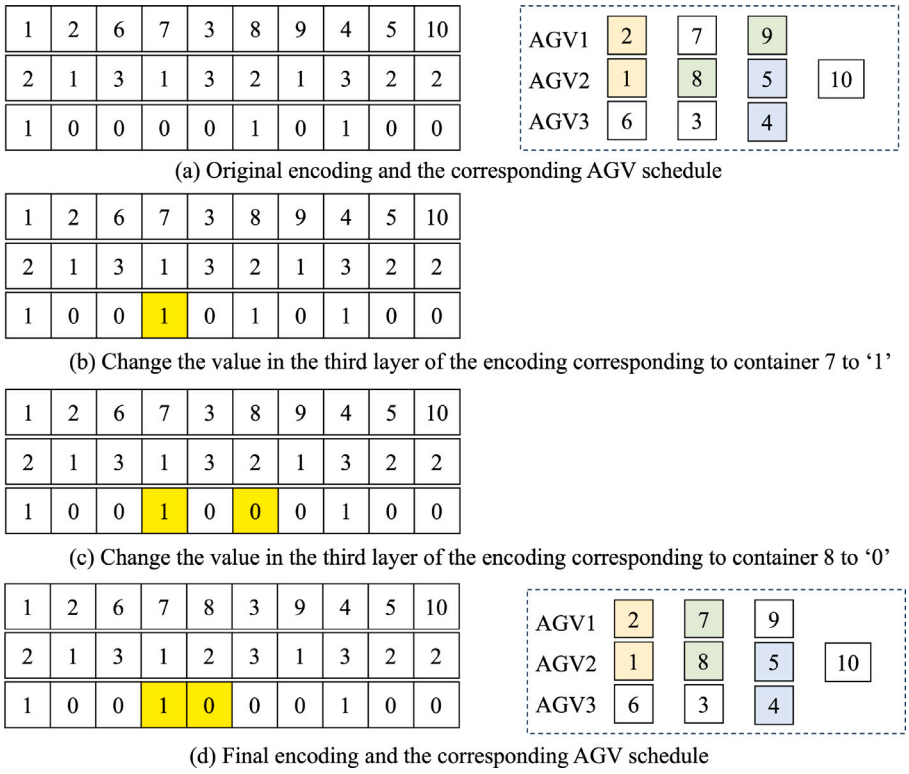


Fig. 13. An example of an operation in NS_3.

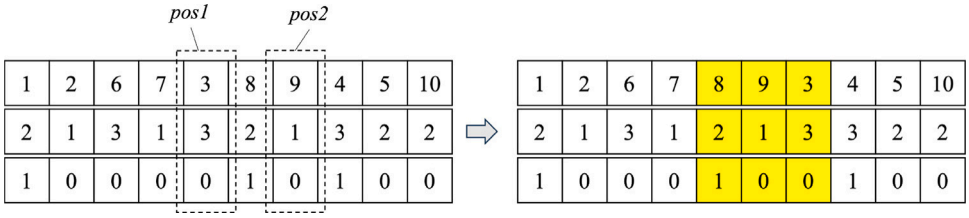


Fig. 14. An example of a feasible exchange in function $Seq_{Exchange}()$.

Table 5
Range of parameters for the computational experiments.

N	Q	V	B	C	K
8~400	1~8	2~20	2~16	1~13	1~3

final encoding and its corresponding AGV schedule are achieved as showcased in Fig. 13(d).

The operations within neighborhood structures NS_k, where $k = 1, 2, 3$, offer the potential to refine the solution by optimizing AGV schedules and TQC operational modes. Due to factors such as AGV congestions and conflicts, different processing sequences for containers under identical AGV schedules and TQC operational modes could lead to varying objective values. Therefore, after obtaining a new solution through the neighborhood search within NS_k, there exists a prospect to further elevate the solution quality by fine-tuning the container processing sequence, all while preserving the AGV allocation and operation sequence schemes, as well as TQC operation modes in an unchanged state. Based on this concept, we propose function $Seq_{Exchange}()$ for adjusting the container processing sequence.

The procedures of function $Seq_{Exchange}()$ are outlined as follows: choose two distinct positions from the encoding, and subsequently interchange all corresponding values associated with these two positions across the three layers, while keeping the corresponding AGV schedule consistent with the original AGV schedule. All the feasible exchanges in function $Seq_{Exchange}()$ should be evaluated, and the best one will be performed.

An example of a feasible exchange in $Seq_{Exchange}()$ is illustrated in Fig. 14. Given the requirement to sustain the adjacency of containers unloaded in a tandem-lift, container 8 is moved in conjunction with container 9 in this scenario.

The exchange in Fig. 15 should be prohibited, as it alters the original AGV schedule.

The function $Seq\_Exchange()$ is invoked every time the best solution within neighborhood NS_k is achieved (as indicated in step 11 of Table 1). If the outcome yielded by $Seq_{Exchange}()$ proves superior, then the solution will be updated accordingly. This procedure helps to further improve the solution during the local search process.

5. Computational experiments

5.1. Instance description

The instances are defined by the following parameters: the number of containers to be unloaded (N); the number of operational TQCs (Q); the number of available AGVs (V); the number of yard blocks (B); the number of transfer buffers at the seaside end of each yard block (C); and the capacity of the AGV path segment (K). Note that, in the

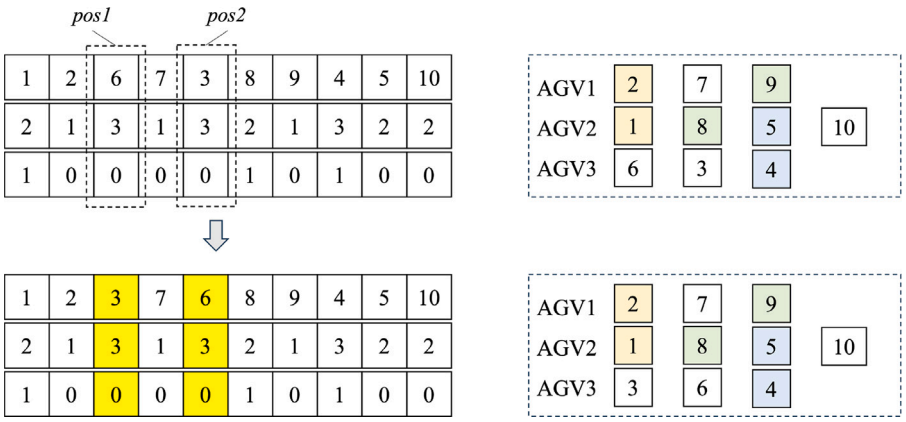


Fig. 15. An example of an infeasible exchange in function $Seq_{Exchange}()$.

Table 6
Computational results of small-scale instances.

No.	Instance#	Gurobi			MS-LS		No.	Instance#	Gurobi			MS-LS	
		Obj	Time (s)	Gap1 (%)	Obj	Time (s)			Obj	Time (s)	Gap1 (%)	Obj	Time (s)
1	8-1-2-2-3-1	87	18.9	0	87	0.1	16	8-1-2-2-3-3	87	3.9	0	87	0.1
2	8-1-4-2-3-1	56	7.9	0	56	0.1	17	8-1-4-2-3-3	56	0.8	0	56	0.1
3	8-1-6-2-3-1	49	2.08	0	49	0.1	18	8-1-6-2-3-3	49	0.6	0	49	0.1
4	10-2-2-2-3-1	N.A.	7200	N.A.	115	0.4	19	10-2-2-2-3-3	115	7200	45.2	115	0.5
5	10-2-4-2-3-1	59	27.16	0	59	0.4	20	10-2-4-2-3-3	59	8.5	0	59	0.2
6	10-2-6-2-3-1	40	2.9	0	40	0.3	21	10-2-6-2-3-3	39	1.38	0	39	0.3
7	12-2-2-2-3-1	N.A.	7200	N.A.	135	0.7	22	12-2-2-2-3-3	N.A.	7200	N.A.	135	0.9
8	12-2-4-2-3-1	67	112.8	0	67	0.6	23	12-2-4-2-3-3	67	51.3	0	67	0.5
9	12-2-6-2-3-1	47	14.26	0	47	0.6	24	12-2-6-2-3-3	47	13.2	0	47	0.5
10	12-3-4-2-3-1	60	408.9	0	60	0.6	25	12-3-4-2-3-3	59	284.9	0	59	0.7
11	12-3-6-2-3-1	49	85.5	0	49	0.8	26	12-3-6-2-3-3	39	492.5	0	39	0.9
12	12-3-8-2-3-1	40	275.1	0	40	0.8	27	12-3-8-2-3-3	31	189.1	0	31	1
13	16-4-6-2-3-1	60	7200	43.3	60	0.9	28	16-4-6-2-3-3	59	7200	40.7	59	1.2
14	16-4-8-2-3-1	49	7200	4.1	49	1.1	29	16-4-8-2-3-3	42	5625.7	0	42	1.4
15	16-4-10-2-3-1	N.A.	7200	N.A.	40	1.3	30	16-4-10-2-3-3	N.A.	7200	N.A.	39	1.4
Average					63.5	0.6	**Average**					61.5	0.7

Note: Instance # denotes N-Q-V-B-C-K.

experiments, we set the capacities of all path segments to be equal. However, our model and algorithm can also be applied to scenarios where path segments possess varying capacities.

The ranges of the above parameters are presented in Table 5, combined by which, small-scale and large-scale instances were generated. The purpose of generating small-scale instances was to validate the consistency of the MILP model and the MS-LS algorithm. On the other hand, large-scale instances were created to assess the effectiveness of the MS-LS algorithm in solving real-scale instances in the ACT. In alignment with the actual conditions in the ACT, the time required for executing a tandem-lift spans from 2 to 3.5 min, while the time for performing a single-lift ranges from 1.7 to 2.9 min, depending on the container positions on the containership. The duration an AGV spends at the AGV-mate is set to 0.5 min. Additionally, the ASC processing time and extra time caused by ASC interactions for a container is generated according to the container's location within the yard block.

5.2. Performance of the MS-LS algorithm

To validate the performance of the proposed MS-LS algorithm, small-scale instances were generated. The results of the small-scale instances were obtained both by solving the MILP model using Gurobi and the MS-LS algorithm. The results obtained by the two methods are presented in Table 6. The column "Instance#" in Table 6 describes the instance settings; the column "Obj" shows the objective value of the instance; the column "Time" shows the computational time of the methods in seconds; the column "Gap1" shows the gap between the best solution and the lower bound obtained by the Gurobi within 7200 s; "N.A." is used to indicate that the Gurobi cannot find any feasible solution within 7200 s.

To better address factors such as AGV congestion and conflicts, the concept of 'time unit' has been incorporated into the model, as well as the heuristic algorithm. The length of a time unit can be adjusted according to practical requirements. In this experiment, in order to ensure the computational efficiency of the model, a time unit of 10 s is set.

In Table 6, when the value of Gap1 equals 0, it represents that the corresponding instance was optimally solved by the Gurobi within the time limit. From Table 6, we can observe that the objective values obtained by the MS-LS algorithm are the same as the results solved by the Gurobi for all the instances that are optimally solved. When the objective values associated with the instances are relatively large, Gurobi is unable to find any feasible solutions within a time limit of 7200 s, such as instances 4, 7, 19, and 22. In contrast, the MS-LS algorithm demonstrates the capability to solve all small-scale instances with high quality within a mere 2 s. The outcomes listed in Table 6 underscore the effectiveness and efficiency of the proposed MS-LS algorithm for small-scale instances.

Furthermore, as indicated in Table 6, the average objective value for instances where the AGV path capacity is 1 (i.e., No. 1–15) is 63.5, whereas the average objective value for instances with an AGV path capacity of 3 (i.e., No. 16–30) is 61.5. These outcomes underscore the substantial influence of the AGV path capacity on the unloading operations.

To further evaluate the effectiveness of the proposed MS-LS algorithm in realistic instances, medium- and large-scale instances were generated according to the rules in Section 5.1. Each instance is solved

Table 7
Computational results of medium- and large-scale instances.

No	Instance#	Obj_{init}	Obj_{max}	Obj_{min}	Obj_{avg}	Gap2 (%)	Gap3 (%)	$Time_{avg}$ (s)
1	100-2-4-4-3-1	971	907	903	905	0.4	7.3	3.0
2	100-2-6-4-3-1	720	650	648	649	0.3	10.9	2.2
3	100-2-8-4-3-1	585	547	543	544	0.7	7.5	3.7
4	100-5-10-4-3-1	398	371	368	369	0.8	7.9	3.8
5	100-5-12-4-3-1	335	312	312	312	0.0	7.4	3.1
6	100-5-14-4-3-1	324	288	286	287	0.7	12.9	3.9
7	200-2-4-8-3-1	2995	2784	2770	2773	0.5	8.0	6.4
8	200-2-6-8-3-1	2122	1935	1924	1927	0.6	10.1	6.2
9	200-2-8-8-3-1	1592	1481	1469	1472	0.8	8.2	7.4
10	200-5-10-8-3-1	1115	1059	1048	1051	1.0	6.1	7.5
11	200-5-12-8-3-1	935	867	858	860	1.0	8.7	7.2
12	200-5-14-8-3-1	799	718	713	715	0.7	11.7	10.6
13	300-3-6-12-3-1	2745	2590	2561	2569	1.1	6.9	26.4
14	300-3-8-12-3-1	2575	2379	2357	2361	0.9	9.1	28.4
15	300-3-10-12-3-1	2311	2040	2021	2026	0.9	14.1	24.9
16	300-6-12-12-3-1	1334	1229	1215	1219	1.2	9.4	28.7
17	300-6-14-12-3-1	1249	1157	1146	1149	1.0	8.7	31.6
18	300-6-16-12-3-1	1152	1039	1031	1033	0.8	11.5	34.2
19	400-4-8-16-3-1	3025	2851	2834	2839	0.6	6.6	47.8
20	400-4-10-16-3-1	2671	2381	2360	2367	0.9	12.8	45.5
21	400-4-12-16-3-1	2469	2310	2286	2291	1.0	7.8	49.3
22	400-8-16-16-3-1	1779	1606	1592	1596	0.9	11.5	48.6
23	400-8-18-16-3-1	1539	1441	1424	1428	1.2	7.8	53.1
24	400-8-20-16-3-1	1425	1324	1307	1313	1.3	8.5	54.2
Average		1549	1428	1416	1419	0.8	9.2	22.4

10 times; for each instance, we calculate the average objective value of the initial solutions (Obj_{init}), the maximum objective value of the final solutions in 10 times (Obj_{max}), the minimum objective value of the final solutions in 10 times (Obj_{min}), the average objective value of the final solutions (Obj_{avg}), the gap between the minimum and maximum objective values (Gap2), the gap between the average objective values of initial and final solutions (Gap3), and the average computational time of the algorithm ($Time_{avg}$). The results are listed in Table 7.

As can be seen in Table 7, the MS-LS algorithm can improve the initial solutions with an average rate of 9.2%, and the average computational time is about 22.4 s, which verifies that the MS-LS algorithm is efficient for solving medium- and large-scale instances. Moreover, by comparing the results in the columns Obj_{max}, Obj_{min}, and Obj_{min}, we found that the solution quality of the MS-LS algorithm is stable for solving the medium- and large-scale instances, because the average gap between the values in column Obj_{max} and column Obj_{min} is only about 0.8%.

5.3. Impact for considering the AGV congestion and conflicts

To further verify the significance of considering AGV congestion and conflicts in the ACTs, an analysis was conducted for two scenarios: one without considering AGV congestion and conflicts and the other considering AGV congestion and conflicts (with a path capacity of 1). Medium-scale instances were generated with AGV quantities ranging from 2 to 20. The MS-LS algorithm was employed to calculate the objective value for each instance under the two scenarios. The calculation results are shown in Fig. 16.

The graph on the left side of Fig. 16 illustrates the variation in the objective value (i.e., the makespan to complete unloading operations) as the number of AGVs changes under two scenarios: one considering AGV congestion and conflicts and the other ignoring AGV congestion and conflicts. It can be observed that, in both scenarios, the objective value decreases as the number of AGVs increases. Notably, the objective value is higher when AGV congestion and conflicts are taken into consideration.

Gap4 represents the ratio of the difference in objective values between these two scenarios. When the number of AGVs is small, the gap between the two scenarios is minor. This implies that the impact of traffic congestion and conflicts on AGV scheduling outcomes is relatively limited in such cases. However, as the number of AGVs increases, the difference in objective values between the two scenarios becomes more pronounced. These findings suggest that, as the number of AGVs increases, factors like AGV congestion and conflicts become increasingly significant and cannot be overlooked during the scheduling process.

In the instances provided, the minimum number of AGVs is 2. The graph on the right side of Fig. 16 provides a visual depiction of the percentage by which the objective value decreases as the number of AGVs increases, compared to the instance with 2 AGVs. This specific percentage is labeled as Gap5. As shown in the figure, the degree of reduction in the objective value achieved by continuously increasing the number of AGVs will gradually diminish, especially when considering AGV congestion and conflicts. Therefore, blindly increasing the number of AGVs to enhance the operational efficiency of an ACT is not advisable, particularly when taking into account factors such as AGV congestion and conflicts. The approach presented in this study can also serve as a simulation method to provide a basis for making decisions on the number of AGVs to be deployed under different circumstances.

5.4. Impact of the number of transfer buffers at the seaside end of the yard block

To further demonstrate the necessity for setting buffer areas at the seaside end of yard blocks, as well as the significance of the consideration of limited yard buffers in this study, we took a sensitivity analysis on the number of transfer buffers. Medium-scale instances were generated: the number of AGVs is set to 6, 8, 10, 12, and 14; the number of yard blocks is set to 4 or 8; the number of transfer buffers varies from 0 to 13. When the number of transfer buffers equals 0, it implies that no transfer buffer is set and the AGVs and ASCs interact directly. The computational results are shown in Fig. 17.

As can be seen from Fig. 17, when the number of transfer buffers is one, the objective value is the largest for all the instances, it is even larger than the situation when no transfer buffer is set. The reason is that, when using the AGV-mate, additional time should be spent on the AGV-mate, and when there is only one transfer buffer, the AGV waiting time using the AGV-mate may be greater than directly interacting with the ASC. Thus, it is meaningless to set only one transfer buffer/AGV-mate in front of the yard block. However, when the number of transfer

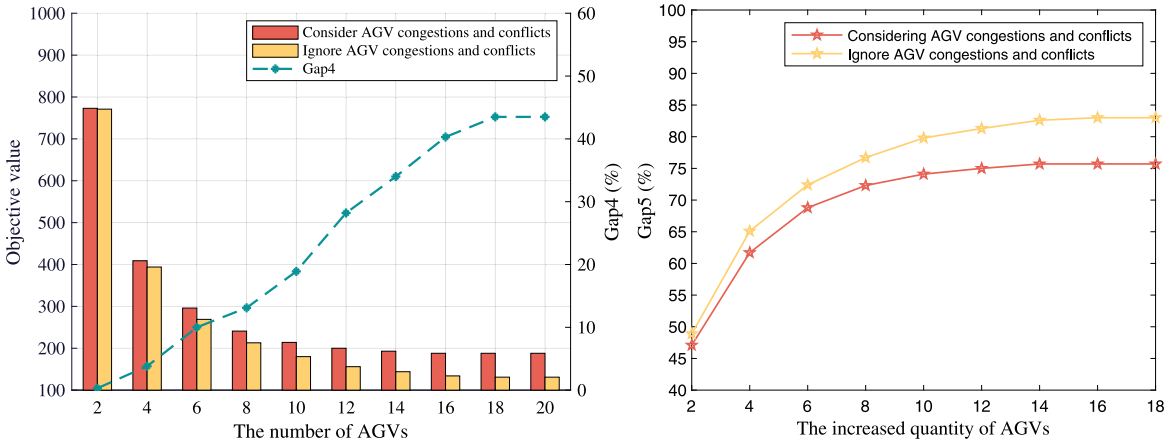


Fig. 16. Calculation results when considering and ignoring AGV congestion and conflicts.

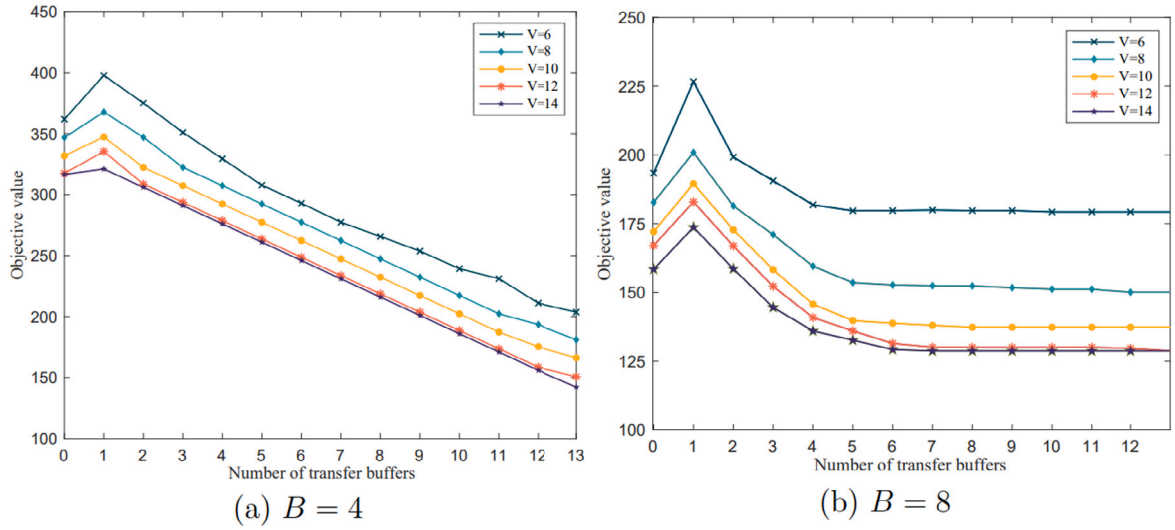


Fig. 17. The impact of the number of transfer buffers on the objective value.

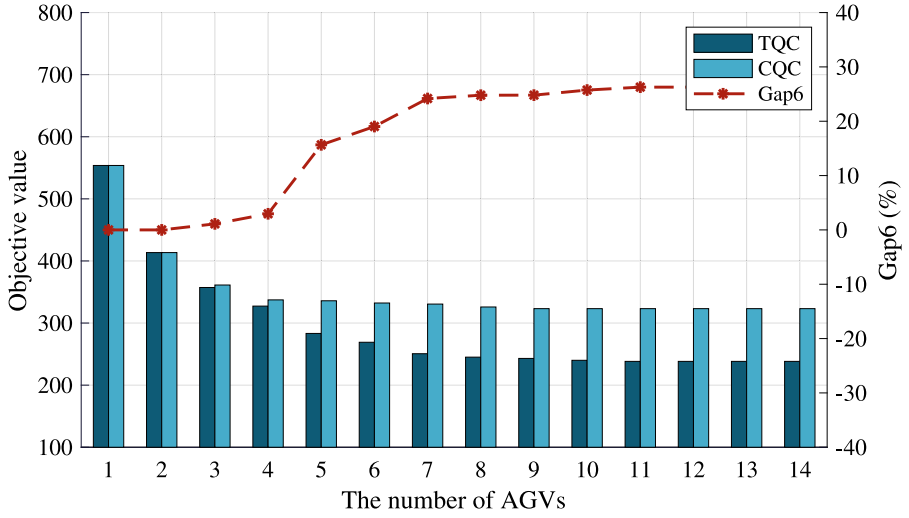


Fig. 18. The completion time for unloading containers using TQCs and CQCs.

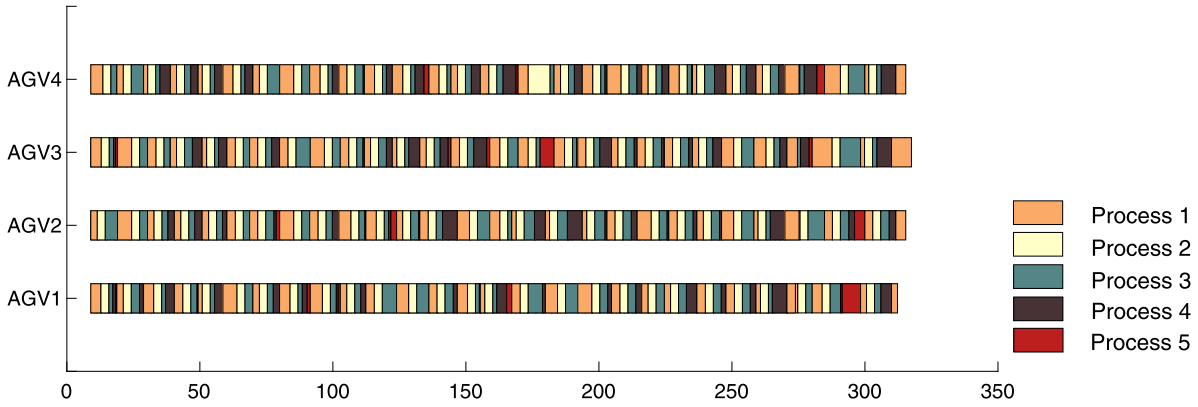


Fig. 19. Time axis of AGVs in the solution of the instance when $V = 4$.

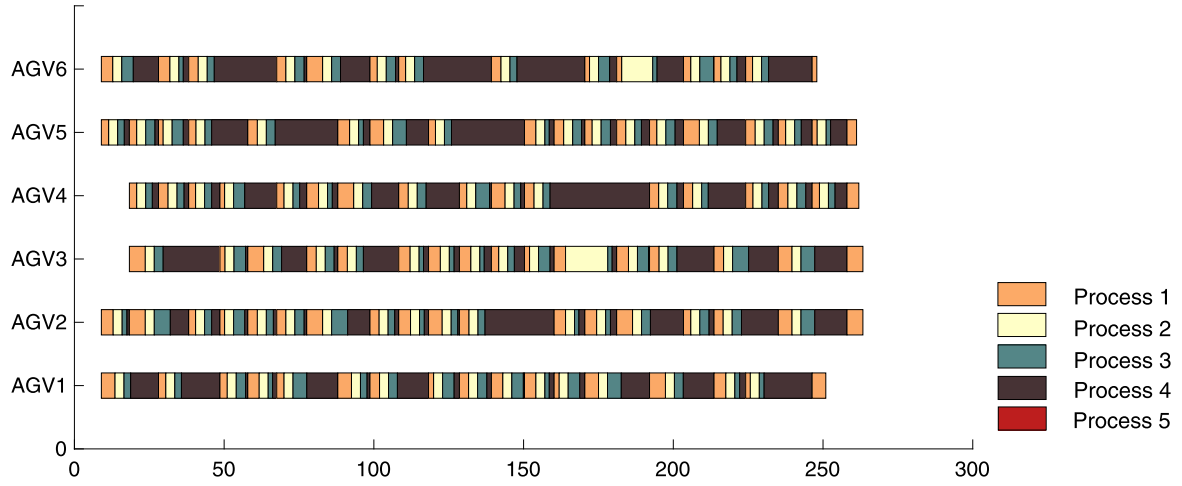


Fig. 20. Time axis of AGVs in the solution of the instance when $V = 6$.

buffers is more than two, the objective value is smaller than in the situation where no transfer buffer is set for all the instances; besides, the objective value tends to decrease with the increasing number of transfer buffers. The above results demonstrate the necessity for setting buffer areas at yard blocks. It indeed promotes operational efficiency at the yard by forbidding direct interactions between ASCs and AGVs.

Because of the limited space and financial resources, only a limited number of transfer buffers can be set at each yard block. Fig. 17 shows that the completion time (objective value) is highly related to the transfer buffers. Thus, the number of transfer buffers must be taken into consideration when scheduling AGVs, which verifies the significance of considering the limited yard buffers in this study.

Moreover, in Fig. 17(a), the objective value decreases as the number of transfer buffers increases; however, in Fig. 17(b), when the number of transfer buffers reaches a certain value, the objective value no longer decreases. The difference between the instances in Fig. 17(a) and (b) is the number of yard blocks. When the number of yard blocks is larger, the number of containers that are allocated to a certain block tends to be smaller. The above results show that when the yard block is not very busy, blindly increasing the number of transfer buffers does not further increase the operational efficiency. It is necessary to set a reasonable number of transfer buffers according to the container arrival rate at the yard blocks. The algorithm in this paper can not only optimize the AGV scheduling scheme under the current equipment layout but also provide a simulation method for the ACT to determine the optimal number of transfer buffers and AGVs.

5.5. Time savings by using the TQCs than the CQCs

A CQC can lift only one 40 ft container while the TQC can lift either one 40 ft container or two 40 ft containers simultaneously. In this section, we testify how much time can be saved by using the TQCs rather than the CQCs under different numbers of AGVs.

The number of containers is set to 100; the number of quay cranes is set to 5; the number of AGVs varies from 1 to 14. We obtained the solutions using the MS-LS algorithm when CQC or TQC is applied. The computational results are shown in Fig. 18.

In Fig. 18, the left Y-axis shows the completion time of unloading operations by the TQCs (i.e., the objective value) while the right Y-axis shows the differences between the completion time of unloading operations using TQCs and CQCs in percentage (Gap6). As seen in Fig. 18, when the number of AGVs is relatively small, the completion time for unloading containers using CQCs is nearly the same as that using TQCs; however, as the number of AGVs continues to rise, the gap between the two scenarios will increase. This confirms that when the TQCs are applied at the ACT, more AGVs should be allocated for serving the TQCs. From Fig. 18, we can conclude that with a sufficient number of AGVs, the TQC is capable of reducing the completion time for unloading operations by approximately 30% compared to the CQC.

For a more intuitive display, we draw the time axis of AGVs in the solution of one case of instance when the number of AGVs is 4 or 6 for serving the TQCs, the results are shown in Figs. 19 and 20, respectively.

In Figs. 19 and 20, "Process 1" represents the AGV traveling time from the quayside to the yard (loaded traveling); "Process 2" represents the AGV waiting time at the yard; "Process 3" represents the AGV traveling time from the yard to the quayside (empty traveling); "Process 4" represents the AGV waiting time for the TQC; "Process 5" represents the AGV waiting time for another AGV for serving a tandem-lift.

From Figs. 19 and 20, we can observe that when the number of AGVs is 4 (relatively scarce), the waiting time between AGVs (Process 5) always exists. However, when the number of AGVs is 6 (relatively enough), the waiting time between AGVs (Process 5) is 0 (or almost 0) by applying our scheduling method.

6. Conclusions

This study investigates the AGV scheduling problem for serving a new type of quay crane, i.e., the TQC, in the ACTs, where traffic congestion and conflicts are taken into consideration. We formulate a MILP model to schedule the AGVs for serving TQCs with the objective of minimizing the completion time of unloading operations by the TQCs. The transfer of containers between AGVs and ASCs is through the AGV-mates at the seaside end of yard blocks to avoid direct interactions between AGVs and ASCs. To be more practical, the limited yard buffers (AGV-mates) at the yard block are considered. Then, a heuristic algorithm (called MS-LS algorithm) is developed to solve the problem based on the VND framework with specific local search procedures.

In future work, the bidirectional flow of containers could be considered in the ACT to reduce the empty traveling of AGVs.

CRediT authorship contribution statement

Lingrui Kong: Conceptualization, Methodology, Software, Writing – original draft, Investigation, Data curation. **Mingjun Ji:** Supervision, Project administration, Writing – review & editing, Funding acquisition. **Anxu Yu:** Validation, Formal analysis. **Zhendi Gao:** Validation, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by the National Nature Science Foundation of China [grant numbers 71971035, 71572022]; and the Cultivation Program for the Excellent Doctoral Dissertation of Dalian Maritime University.

Appendix. Method for estimating the extra ASC processing time caused by ASC interactions within a block.

The seaside ASC is denoted as ASC1, while the landside ASC is referred to as ASC2. The following parameters are introduced:

N: the number of bays in a yard block. The bays in a block are sequentially numbered from the seaside to the landside as $1, 2, \dots, N$.

n_i: the storage bay of container i in the yard block, $i \in \Omega$.

ω: the travel time for the ASC to move through a bay.

γ: the expected ASC2 operational time within a bay for completing a loading/unloading task.

n': the storage bay of an ASC2 task. Assuming that containers are randomly stored in a yard block, we have $P(n') = \frac{1}{N}$, where $n' = 1, 2, \dots, N$. Here, $P(n')$ represents the probability that an ASC2 task is stored in bay n'.

Assume that the two ASCs have identical moving speeds between bays. In the event that two ASCs will conflict at a specific bay, the following rules apply: (1) The ASC that arrives at the designated bay later should wait. (2) If the two ASCs will reach their designated bays simultaneously, ASC2 should be the one to wait.

The extra ASC processing (waiting) time caused by ASC interactions within a block is closely tied to container storage positions within the yard block and the arrival rates of AGVs and external trucks at the block. First, considering the influence of container storage positions, we can derive the waiting time of ASC1 caused by ASC interactions as follows.

If $n_i = 1$, conflicts between the two ASCs will only occur when $n' = 1$,

(1) When $n' = 1$, the worst-case scenario for ASC1 unfolds as follows: just as ASC1 is about to reach bay 1, ASC2 has already arrived at bay 1. In this situation, ASC1 must wait, and the waiting time should be $\omega + \gamma$.

(2) When $n' = 2, 3, \dots, N$, the waiting time of ASC1 is 0.

Therefore, the average waiting time of ASC1 when $n_i = 1$ is given by $E(n_i = 1) = P(1) \times (\omega + \gamma) = \frac{\omega+\gamma}{N}$.

If $n_i = 2$, then,

(1) When $n' = 1$, the worst-case scenario for ASC1 unfolds as follows: just as ASC1 is about to reach bay 1, ASC2 has arrived at bay 2. In this situation, ASC1 must wait, and the waiting time should be $2\omega + \gamma$.

(2) When $n' = 2$, the worst-case scenario for ASC1 unfolds as follows: just as ASC1 is about to reach bay 2, ASC2 has arrived at bay 2. In this situation, ASC1 must wait, and the waiting time should be $\omega + \gamma$.

(3) When $n' = 3, \dots, N$, the waiting time of ASC1 should be 0.

Thus, the average waiting time of ASC1 when $n_i = 2$ is given by $E(n_i = 2) = P(1) \times (2\omega + \gamma) + P(2) \times (\omega + \gamma) = \frac{3\omega+2\gamma}{N}$.

Continuing in a similar manner, we can calculate $E(n_i = 3) = \frac{(3+2+1)\omega+3\gamma}{N}$, and $E(n_i = 4) = \frac{(4+3+2+1)\omega+4\gamma}{N}$.

To be more general, $E(n_i = n) = \frac{(n+n-1+\cdots+1)\omega+n\gamma}{N} = \frac{n\omega(n+1)/2+n\gamma}{N}$.

Then, by incorporating the impact of arrival rates of AGVs and external trucks, we use the following formula to estimate the extra waiting time of ASC1 caused by ASC interactions within a block.

$$d_i = \alpha \times \frac{n_i\omega(n_i+1)/2 + n_i\gamma}{N}, \tag{45}$$

where α is a decimal value between 0 and 1. Its significance is closely tied to the arrival rates of AGVs and external trucks at a container block. When both the AGV and external truck arrival rates are notably elevated, the likelihood of congestion occurring between two ASCs increases. Consequently, the value of α is adjusted to a larger magnitude. Conversely, when the arrival rates of AGVs and external trucks are relatively low, the probability of congestion diminishes, and, as a result, the value of α is set to a relatively smaller value.

By employing Eq. (45), we can estimate the waiting time of ASC1 due to conflicts with ASC2 for each individual container. By integrating this time into the scheduling process, the resulting schedules will exhibit greater resilience to the practical operational conditions.

References

Angeloudis, P., Bell, M.G.H., 2010. An uncertainty-aware AGV assignment algorithm for automated container terminals. Transp. Res. E 46 (3), 354–366.

Bartošek, A., Marek, O., 2013. Quay cranes in container terminals. Trans. Transp. Sci. 6 (1), 9–18.

Bezerra, S., Souza, M., Souza, S., 2022. A variable neighborhood search-based algorithm with adaptive local search for the vehicle routing problem with time windows and multi-depots aiming for vehicle fleet reduction. Comput. Oper. Res. 149, 106016. http://dx.doi.org/10.1016/j.cor.2022.106016.

Bian, Z., Yang, Y., Mi, W., Mi, C., 2015. Dispatching electric agvs in automated container terminals with long travelling distance. J. Coastal Res. 73, 75–81.

Bierwirth, C., Meisel, F., 2010. A survey of berth allocation and quay crane scheduling problems in container terminals. Eur. J. Oper. Res. 202 (3), 615–627.
Bierwirth, C., Meisel, F., 2015. A follow-up survey of berth allocation and quay crane scheduling problems in container terminals. Eur. J. Oper. Res. 244 (3), 675–689.
Boysen, N., Briskorn, D., Meisel, F., 2017. A generalized classification scheme for crane scheduling with interference. Eur. J. Oper. Res. 258 (1), 343–357.
Chao, S.L., Lin, Y.J., 2011. Evaluating advanced quay cranes in container terminals. Transp. Res. E 47 (4), 432–445.
Chen, L.H., Cao, J.X., Zhao, Q.Y., 2014. Tandem lift quay cranes and yard trucks scheduling problem at container terminals. Appl. Mech. Mater. 505–506, 927–930.
Chen, X., He, S., Zhang, Y., Tong, L.C., Shang, P., Zhou, X., 2020. Yard crane and AGV scheduling in automated container terminal: A multi-robot task allocation framework. Transp. Res. C 114, 241–271.
Choe, R., Kim, J., Ryu, K.R., 2016. Online preference learning for adaptive dispatching of AGVs in an automated container terminal. Appl. Soft Comput. 38, 647–660.
Choi, S.H., Im, H., Lee, C., 2014. Development of an operating system for optimization of the container terminal by using the tandem-lift quay crane. Lect. Notes Electr. Eng. 276 (5), 399–404.
Christof, D., Kenneth, S., 2017. A fast two-level variable neighborhood search for the clustered vehicle routing problem. Comput. Oper. Res. 83, 78–94.
Drungilas, D., Kurmis, M., Senulis, A., Lukosius, Z., Andziulis, A., Januteniene, J., Bogdevicius, M., Jankunas, V., Voznak, M., 2023. Deep reinforcement learning based optimization of automated guided vehicle time and energy consumption in a container terminal. Alex. Eng. J. 67, 397–407.
Duan, J., Li, L., Zhang, Q., Qin, J., 2023. Integrated scheduling of automatic guided vehicles and automatic stacking cranes in automated container terminals considering landside buffer zone. Transp. Res. Rec. http://dx.doi.org/10.1177/03611981231168862.
Grunow, M., Gunther, H.O., Lehmann, M., 2006. Strategies for dispatching AGVs at automated seaport container terminals. OR Spectrum 28 (4), 587–610.
Guo, X.H., Ji, M.J., Zhang, W.D., 2023. Research on a new two-level scheduling approach for unmanned surface vehicles transportation containers in automated terminals. Comput. Ind. Eng. 175, 108901. http://dx.doi.org/10.1016/j.cie.2022.108901.
Hu, H., Chen, X., Wang, T., Zang, Y., 2019. A three-stage decomposition method for the joint vehicle dispatching and storage allocation problem in automated container terminals. Comput. Ind. Eng. 129, 90–101.
Huang, S.Y., Li, Y., 2017. Optimization and evaluation of tandem quay crane performance. In: The 2017 4th International Conference on Systems and Informatics, ICSAI, Hangzhou. pp. 637–642.
Jonker, T., Duinkerken, M.B., Yorke-Smith, N., Waal, A., Negenborn, R.R., 2021. Coordinated optimization of equipment operations in a container terminal. Flex. Serv. Manuf. J. 33 (2), 281–311.
Kim, K.H., Bae, J.W., 2004. A look-ahead dispatching method for automated guided vehicles in automated port container terminals. Transp. Sci. 38 (2), 224–234.
Kong, L.R., Ji, M.J., Gao, Z.D., 2021. Joint optimization of container slot planning and truck scheduling for tandem quay cranes. Eur. J. Oper. Res. 293 (1), 149–166.
Kong, L.R., Ji, M.J., Gao, Z.D., 2022. An exact algorithm for scheduling tandem quay crane operations in container terminals. Transp. Res. E 168, 102949. http://dx.doi.org/10.1016/j.tre.2022.102949.
Kytoejoki, J., Nuortio, T., Braeysy, O., Gendreau, M., 2007. An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. Comput. Oper. Res. 34 (9), 2743–2757.
Lashkari, S., Wu, Y., Petering, M., 2017. Sequencing dual-spreader crane operations: Mathematical formulation and heuristic algorithm. Eur. J. Oper. Res. 262 (2), 521–534.
Lau, H.Y., Zhao, Y., 2008. Integrated scheduling of handling equipment at automated container terminals. Ann. Oper. Res. 159 (1), 373–394.
Liu, D., Ge, Y.E., 2018. Modeling assignment of quay cranes using queueing theory for minimizing CO2 emission at a container terminal. Transp. Res. D 61A, 140–151.
Liu, W., Zhu, X., Wang, Li, Wang, S., 2023. Multiple equipment scheduling and AGV trajectory generation in U-shaped sea-rail intermodal automated container terminal. Measurement 206, 112262. http://dx.doi.org/10.1016/j.measurement.2022.112262.
Luo, J., Wu, Y., 2015. Modelling of dual-cycle strategy for container storage and vehicle scheduling problems at automated container terminals. Transp. Res. E 79, 49–64.
Luo, J., Wu, Y., Mendes, A.B., 2016. Modelling of integrated vehicle scheduling and container storage problems in unloading process at an automated container terminal. Comput. Ind. Eng. 94, 32–44.
Meersmans, P.J.M., Wagelmans, A.P.M., 2001. Effective Algorithms for Integrated Scheduling of Handling Equipment at Automated Container Terminals. ERIM Report Series Reference No. ERS-2001-36-LIS, Available at SSRN: https://ssrn.com/abstract=370889.
Mladenovi, N., Hansen, P., 1997. Variable neighborhood search. Comput. Oper. Res. 24 (11), 1097–1100.
Nguyen, V.D., Kim, K.H., 2009. A dispatching method for automated lifting vehicles in automated port container terminals. Comput. Ind. Eng. 56 (3), 1002–1020.
Roy, D., Koster, R.D., Bekker, R., 2020. Modeling and design of container terminal operations. Oper. Res. 68 (3), 686–715.
Sadati, M., Atay, B., 2021. A hybrid variable neighborhood search approach for the multi-depot green vehicle routing problem. Transp. Res. E 149 (4), 102293.
Sadeghian, S.H., Ariffin, M.K.A.M., Tang, S.H., Ismail, N., 2014. Integrated dispatching model of automated lifting vehicles, quay cranes and yard cranes at automated container terminal. Appl. Mech. Mater. 564, 678–683.
Stenger, A., Enz, S., Schwind, M., Vigo, D., 2014. An adaptive variable neighborhood search algorithm for a vehicle routing problem arising in small package shipping. Oper. Res. 54 (1–2), 85–87.
Sun, P.Z., You, J., Qiu, S., Wu, E.Q., Xiong, P., Song, A., Zhang, H., Lu, T., 2022. AGV-based vehicle transportation in automated container terminals: a survey. IEEE Trans. Intell. Transp. Syst. 24 (1), 341–356.
Tommaso, A., Tolga, B., Gianpaolo, G., Emanuela, G., Emanuele, M., 2018. Path and speed optimization for conflict-free pickup and delivery under time windows. Transp. Sci. 52 (4), 739–755.
UNCTAD, 2022. Review of maritime transport 2022. In: United Nations Conference on Trade and Development.
Wang, Z., Zeng, Q., 2022. A branch-and-bound approach for AGV dispatching and routing problems in automated container terminals. Comput. Ind. Eng. 166, 107968. http://dx.doi.org/10.1016/j.cie.2022.107968.
Xin, J., Meng, C., D'Ariano, A., Wang, D., Negenborn, R.R., 2022. Mixed-integer nonlinear programming for energy-efficient container handling: Formulation and customized genetic algorithm. IEEE Trans. Intell. Transp. Syst. 23 (8), 10542–10555.
Xin, J., Negenborn, R.R., Corman, F., Lodewijks, G., 2015. Control of interacting machines in automated container terminals using a sequential planning approach for collision avoidance. Transp. Res. C 60, 377–396.
Xin, J., Negenborn, R.R., Lodewijks, G., 2014. Energy-aware control for automated container terminals using integrated flow shop scheduling and optimal control. Transp. Res. C 44, 214–230.
Xing, Z., Liu, H., Wang, T., Chew, E.P., Lee, L.H., Tan, K.C., 2023. Integrated automated guided vehicle dispatching and equipment scheduling with speed optimization. Transp. Res. E 169, 102993.
Xing, Y., Yin, K., Quadrifoglio, L., Wang, B., 2012. Dispatch problem of auto-mated guided vehicles for serving tandem lift quay crane. Transp. Res. Rec. 2273 (1), 79–86.
Xu, Y.X., Qi, L., Luan, W.J., Guo, X.W., Ma, H.J., 2020. Load-in-load-out agv route planning in automatic container terminal. IEEE Access 8, 157081-157088.
Yang, Y., Zhong, M., Dessouky, Y., Postolache, O., 2018. An integrated scheduling method for AGV routing in automated container terminals. Comput. Ind. Eng. 126, 482–493.
Yin, Y.Q., Zhong, M.S., Wen, X., Ge, Y.E., 2023. Scheduling quay cranes and shuttle vehicles simultaneously with limited apron buffer capacity. Comput. Oper. Res. 151, 106096.
Zhang, Q., Hu, W., Duan, J., Qin, J., 2021. Cooperative scheduling of AGV and ASC in automation container terminal relay operation mode. Math. Probl. Eng. 2021, 1–18.
Zheng, X., Liang, C., Wang, Y., Shi, J., Lim, G., 2022. Multi-AGV dynamic scheduling in an automated container terminal: a deep reinforcement learning approach. Mathematics 10 (23), 4575.
Zhong, Z., Guo, Y., Zhang, J., Yang, S., 2023. Energy-aware integrated scheduling for container terminals with conflict-free AGVs. J. Syst. Sci. Syst. Eng. 32 (4), 413–443.
Zhong, M., Yang, Y., Dessouky, Y., Postolache, O., 2020. Multi-AGV scheduling for conflict-free path planning in automated container terminals. Comput. Ind. Eng. 142, 106371. http://dx.doi.org/10.1016/j.cie.2020.106371.
Zhong, M., Yang, Y., Zhou, Y., Postolache, O., 2019. Adaptive autotuning mathematical approaches for integrated optimization of automated container terminal. Math. Probl. Eng. 2019 (4), 1–14.
Zhuang, Z., Zhang, Z., Teng, H., Qin, W., Fang, H., 2022. Optimization for integrated scheduling of intelligent handling equipment with bidirectional flows and limited buffers at automated container terminals. Comput. Oper. Res. 145, 105863. http://dx.doi.org/10.1016/j.cor.2022.105863.