Optimizing Warehouse Operations with Autonomous Mobile Robots

Lu Zhen [1], Zheyi Tan [1], René de Koster [2], Xueting He [1], Shuaian Wang [3], Huiwen Wang [1]

[1] *School of Management, Shanghai University, Shanghai, China*
[2] *Rotterdam School of Management, Erasmus University, The Netherlands*
[3] *Faculty of Business, The Hong Kong Polytechnic University, Kowloon, Hong Kong*

Abstract: Autonomous mobile robots (AMRs) can support human pickers in warehouse picking operations by reducing picker walking distance and increasing the warehouse's throughput. AMR-assisted order picking is becoming popular as it can be conveniently implemented in conventional warehouses. This study proposes an integrated optimization model for scheduling the operations in AMR-assisted picker-to-parts warehouse systems. The model aims to minimize the makespan of all picking operations for a batch of orders by assigning batched orders to AMRs, selection of storage racks for AMRs and pickers to visit, and determining the routes of the AMRs and the pickers. A column-and-row generation algorithm is designed to solve the model using synchronization constraints between AMRs and pickers. Numerical experiments are conducted to validate the applicability of our proposed algorithm in a warehouse that handles 16,000 orders per day. Our algorithm can solve small-scale instances to optimality. Our algorithm can also obtain better solutions in shorter time than a column generation-based method. Extensive experiments are conducted to derive managerial insights.

Keywords: warehouse intralogistics; autonomous mobile robots; collaborative robots; picker-to-parts systems; column-and-row generation.

1. Introduction

In the era of e-commerce, online retailers are facing the challenge of handling many time-critical orders. Thus, online retailers are increasingly using automated and robotic handling systems in their warehouses to increase the operational efficiency of order picking, which enhances the responsiveness of their service platforms. The autonomous mobile robot (AMR)-assisted order picking system (also called collaborative robot or cobot-picking) is novel and easy-to-implement. It combines the advantages of automated robotic handling equipment and traditional picker-to-parts systems, and it is increasingly used in many warehouses, particularly by e-commerce retailers.

Compared to systems like Kiva and AutoStore, which rely on expensive special racks (Kiva), fixed rack grid structures (Autostore), and predefined workflows, AMRs can operate in any existing warehouse without major investments, next to manual pickers and partly take over their work. They can move freely within a defined space and adapt quickly to changes in warehouse layouts without the need for reprogramming or hardware modifications. This flexibility allows AMR-assisted systems to adapt to changes in operational demand and warehouse environments (Fragapane et al., 2021). Moreover, compared to traditional picker-to-parts systems, which often require substantial manual effort, AMR-

assisted systems can significantly enhance picker productivity. AMRs reduce physical strain on human pickers and streamline picking tasks. In addition, the precise navigation and task execution provided by AMRs help minimize human errors, improving accuracy and overall productivity.

Figure 1 shows different AMRs, used in traditional warehouse layouts with storage racks each storing multiple products. An AMR is computer-controlled and equipped with sensors (Fragapane et al., 2021). It has a display showing pick instructions for the human pickers and it carries one or more order bins, each of which can hold products required by one customer order. A central control system routes the paths for the AMRs so that they visit several designated storage racks along their paths in a sequence. When an AMR arrives at a storage rack on its path, it waits for a human picker to arrive (or the picker has already arrived and is waiting for the AMR). Then, according to the information displayed on the AMR screen, the picker picks the products (stock keeping units, SKUs) required for the orders assigned to the AMR. The picks are confirmed using a barcode scanner, or by confirmation on the screen. When all the required SKUs in a route have been collected, the AMR independently (without a human picker) goes to the packing station where the collected products are packed by order and transferred to the next step (transportation to customers). As pickers can only focus on picking activities, using AMRs considerably reduces the walking distance of pickers and thus increases the picker productivity. In order to realize a short makespan of a given set of orders, the routes of the AMRs and the pickers, as well as their arrival times (or leaving times) at (or from) the target storage racks should be scheduled appropriately, so that waiting times of pickers and robots and delays are minimal.

Figure 1: AMRs and various warehouses with the AMRs

In the AMR-assisted warehouse system, the continuously arriving orders should be assigned properly to the AMRs. Each order contains a list of SKUs, each for a certain number of units ordered by the customer. The weights (or volumes) of a unit of different SKUs are not identical, and the AMR's capacity with respect to the weight (or volume) of the cargo carried should be taken into account. According to the

information on the SKUs (and the number of units of each SKU) stored in each storage rack, the selection of the storage racks for each AMR to visit should also be properly determined so that the SKU requirement of the orders assigned to the AMR can be satisfied by the selected storage racks. Note that ecommerce warehouses often deploy *scattered* storage (Schiffer et al., 2022), where certain quantities of a SKU can be placed at multiple storage locations. The sequence of visiting these storage racks should also be properly determined so that the AMR route length is not too long, and the AMR visits at some of the storage racks can be synchronized with the picker's visits at these storage racks. These intertwined decisions should be made carefully to minimize the total makespan of the picking operations for all of the orders in a batch.

This study formulates a mixed-integer linear programming (MILP) model for the above-mentioned integrated decision problem regarding (batched) order assignment to AMRs, selection of storage racks for the orders (taking into account scattered SKU storage), AMR routing, and picker routing in an AMR-assisted warehouse. The model is much more complex than previous works on assisted order picking, as all relevant decision problems are included: order batching and assignment to the vehicles, rack selection for both the AMRs and the pickers, pick quantity determination (as products are scattered over locations), and picker and AMR routing. It is also much more difficult than the traditional vehicle routing problem (VRP) and its variants because of the synchronization constraints related to the picker and AMR routes, and the selection of visiting locations (storage racks) for two types of "vehicles" (pickers and AMRs) based on SKU matching. To solve the model using realistic-scale instances within a reasonable time, a column-and-row generation algorithm is designed. Furthermore, numerical experiments are also conducted to validate the efficiency of our proposed algorithm and derive some insights for the practitioners who operate AMR-assisted warehouses.

The study makes three main contributions. First, we propose a novel MILP model for AMR-assisted order picking considering a comprehensive set of factors and decisions. Second, this study designs a novel column-and-row generation algorithm that contains two types of columns and pricing problems and two types of submodels for row generation. Our proposed algorithm is more complex than related algorithms in the literature, but efficient and accurate. It may provide guidelines for applying the column-and-row generation in other large-scale application contexts. Third, the model and algorithm generate useful managerial insights. We find that deploying more or faster AMRs reduces the makespan, but the effect gradually decreases as the speed or the number of AMRs increase. The loading capacity of AMRs also affects the performance; when the capacity increases, the makespan is reduced or remains unchanged (beyond a threshold); and the AMR weight capacity has a more significant influence on the makespan than its volume capacity. In addition, this study derives insights on area zoning and layout strategies. For

example, more zones yield a larger makespan. Moreover, layouts with a cross-aisle in the middle of the warehouse, which allow making shortcuts, result in a shorter makespan than layouts without a cross-aisle.

2. Related work

This paper studies an integrated optimization problem on order assignment and robot & human picker routing in AMR-assisted picker-to-parts warehouse systems. The core of the proposed MILP model is related to a VRP with synchronization constraints. A column-and-row generation algorithm is designed for solving the model. This study is related to three streams of literature: optimization problems in human-robot coordinated warehousing systems, VRPs with synchronization constraints, and the applications of column-and-row generation algorithms.

2.1 Human-robot coordinated warehousing systems

Robot technology allows making labor-intensive processes in warehouses more efficient (Boysen et al., 2019; Azadeh et al., 2019). Multiple studies examine parts-to-picker robotic warehouse systems from various perspectives, including warehouse design and configuration (Lamballais et al., 2017), order batching and sequencing (Boysen et al., 2017), and different storage area zoning policies (Roy et al., 2019). However, modeling challenges remain in coordinating human-robot interactions in warehouse operations.

Wang et al. (2022) investigate human-robot coordinated order picking, in a parts-to-picker system where mobile racks are transported by robots to human-operated pick stations. They consider schedule-induced fluctuations in human picker work states and examine the problem of determining an optimal robot schedule, which involves assigning racks to multiple pickers and sequencing these assignments. Lee and Murray (2019) study a picker-to-parts system, where pick robots, replacing human pickers, retrieve products from shelves and transfer them to transport robots. These then deliver the items to packing stations for later shipment. They explore the routing problem involving these wo types of mobile robots and formulate a MILP model to minimize the latest delivery time of all items to the packing station. As their problem does not contain the concept of "orders", they do not consider the order assignment to robots, which differentiates it from our research. Löffler et al. (2021) examine the robot-assisted picking problem under a fixed-assignment policy, simplifying the problem into isolated single-picker routing challenges. They propose a MILP model to minimize total travel distance and design a polynomial-time routing algorithm to solve it. Azadeh et al. (2023) study a similar system, but then using stochastic modeling for arrivals and travel times. Additionally, Žulj et al. (2022) study an AMR-assisted picker-to-parts system by partitioning the warehouse into disjoint zones and assigning an order picker to each zone. In this system, an order picker collects batch items stored within his or her designated zone and brings

them to handover locations, from which the AMR collects the items and brings them to the depot. They develop a MILP model and an efficient heuristic algorithm to solve it. In our study, pickers and AMRs can navigate freely throughout the warehouse, rather than being restricted to specialized locations (such as designated zones or handover locations). This flexibility complicates the overall problem.

Compared to the reviewed literature, our study incorporates a more comprehensive decision-making process, encompassing order assignment, selection of storage racks for each AMR (as products can be dispersed over multiple storage locations), routing for both AMRs and pickers, and scheduling their arrival and departure times at the target storage racks. These interconnected decisions are integrated into our proposed model to minimize the total makespan of picking operations for all orders in a batch.

2.2 VRP with synchronization constraints

The VRP with synchronization constraints (VRPSC) is one of the VRP variants, in which more than one vehicle may or must be used to fulfill a task (Drexl 2012; Liu et al., 2019; Hà et al., 2020). The interdependence between vehicle routes requires different types of synchronization, such as task, load, space and time synchronization. The inclusion of these synchronization constraints significantly increases the complexity of this variant (Luo et al., 2016; Chen et al., 2020).

One class of VRPSC variants is the synchronized VRP with split deliveries, where multiple vehicles can visit a customer, with each vehicle delivering or collecting a portion of the customer's demand or supply (Shao et al., 2019). Gamst et al. (2024) investigate a standard split delivery VRP, focusing on task and load synchronization at customer locations to ensure that each task (i.e., customer service) is completed exactly once by one or more suitable vehicles while adhering to load constraints. Additionally, Li et al. (2020) examine the VRP with split delivery and time windows, where each customer specifies multiple time windows, one of which is selected for delivery synchronization, thus requiring time synchronization to be addressed. Yu et al. (2022) address the periodic VRP with time windows for perishable products, where the order selection and distribution planning in online community group-buying should be synchronized. However, these studies do not consider synchronization between different types of vehicles (i.e., human pickers and robots in our study), which is specifically addressed by us.

Another class of VRPSC variants is the active-passive VRP, where pickup and delivery requests rely on coordinated operations between active and passive vehicles (Meisel and Kopfer, 2014). This variant also supports flexible coupling and decoupling of vehicles at customer locations (Tilk et al., 2018), incorporating time and space synchronization. Fink et al. (2019) further extend the variant by integrating load synchronization in airport ground handling operations, where both workers and vehicles are considered passive. In contrast to these studies, our research involves both pickers and robots as active agents, and their routes need to be synchronized to complete tasks.

Synchronization constraints are also considered in two-echelon VRPs, in which goods are transferred from the first to the second echelon through synchronized meetings between vehicles from the two echelons. Anderluh et al. (2021) introduce a two-echelon VRP and design maximum time span constraints, which are integrated into their proposed MILP model to limit the waiting time for both types of vehicles at the satellites. Different from our problem, these synchronization stations (i.e., satellites) are given rather than determined by their MILP model. Zhou et al. (2023) study a two-echelon VRP with drones, where multiple vehicles and drones work collaboratively to serve customers. They incorporate space and time synchronization at customer locations to ensure drones can only be launched from customers where a vehicle stops. In this problem, a customer can be served only once by a vehicle or a drone, thus load synchronization is not considered.

Our study proposes a new variant of the VRPSC, which encompasses some types of synchronization different from the literature of the VRPSC. Specifically, we determine order assignments to AMRs considering task synchronization between the AMRs and human pickers, and select storage racks for the orders with load synchronization, which means the weight and volume of cargos contained in each order, carried by each capacitated robot are considered in picking tasks assignment and scheduling. Additionally, routing both the AMRs and pickers, which involves space and time synchronization at storage racks, are integrated into the problem. Furthermore, unlike previous studies, the synchronization stations (i.e., racks) are not predetermined but are optimized within our proposed model.

2.3 Column-and-row generation algorithms

The column-and-row generation technique was initially proposed as a generalization of the column-generation method for solving integer programming problems such as the classical cutting stock problem (Zak 2002). Since then, many scholars have applied this technique to a series of studies on variants of the cutting stock problem. Muter and Sezer (2018) use column-and-row generation to solve a two-stage extension of a one-dimensional cutting stock problem, and Wang et al. (2020) further extend it to a two-dimensional problem considering a skiving process. Different from these applications to classical problems, such as the cutting stock problem, some recent studies apply column-and-row generation to more practical problem contexts, such as airline crew pairing problem (Muter et al., 2013b), medical resource allocation problem (Holte and Mannino 2013), railway crew capacity planning problem (Suyabatmaz and Şahin 2015), meal delivery routing problem (Yildiz and Savelsbergh 2019), and network search game (Yolmeh and Baykal-Gürsoy 2021). These studies construct a restricted master problem (RMP) and a pricing subproblem (PSP). A subset of original constraints is considered in RMP; the missing constraints are added along with the addition of newly generated columns. As there is no row-generating subproblem that is formulated explicitly, Holte and Mannino (2013) design an algorithm,

called separation oracle, to generate constraints to be added to the RMP.

Muter et al. (2013a) first highlight the difficulty of solving problems where the number of constraints in the RMP rapidly increases as the number of variables increases. Such problems that motivate simultaneous generation of both columns and their associated linking constraints are called problems with column-dependent-rows. Muter et al. (2013a) propose a column-and-row generation framework with a row-generating PSP that can calculate reduced costs in the absence of some linking constraints. Their method can solve typical problems with column-dependent-rows efficiently, and it paved the way for later studies such as Alfieri et al. (2020) and Wang et al. (2020). In order to solve large-scale instances, Ucar et al. (2017) improve the column-and-row generation by developing a new heuristic for PSP, but this method compromises on exactness. Moreover, other scholars explore the compatibility of column-and-row generation by embedding it into other exact methods. Maher (2016) embeds column-and-row generation in a branch-and-price scheme for solving an integrated airline recovery problem. Muter et al. (2018) further embed column-and-row generation in Benders decomposition.

Compared with the problem in this study, the problems studied in the above column-and-row generation related literature are quite limited in scale and complexity of the problem. Although the column-and-row generation algorithm proposed in this study follows the work of Muter et al. (2013a), our problem and algorithm are oriented to a much larger scale real-world application context. Moreover, our column-and-row generation algorithm contains two pricing problems for generating two different types of columns, and two novel mathematical models for row generation. To our best knowledge, these algorithmic features are novel in the column-and-row generation related literature. In addition, this study elaborates the framework and formulates explicit mathematical models on three types of essential problems (short RMP, column-generating PSPs, and row-generating submodels) inside the column-and-row generation, which can provide more detailed guidelines for further extensions and applications of column-and-row generation in other large-scale and real-world applications.

3. Problem description

Suppose there is a warehouse with a set of AMRs and a set of pickers. Hereinafter, AMRs are referred to as “robots.” Given the layout of storage racks in the warehouse, the travel time for a robot (or a picker) from its current location to a storage rack, or between two storage racks, is known. Orders are continuously entered into the warehouse management system (WMS). For each order, the list of SKUs contained in the order and the number of units required for each SKU is known. In addition, the WMS stores the information on the weight or volume of one unit of each SKU, the maximum weight or volume of cargo carried by a robot, the maximum number of orders assigned to a robot, and the number of units

of an SKU stored in each storage rack (with multiple SKUs per rack). The WMS must make decisions at a certain frequency (e.g., every 3 minutes). These decisions include assigning each order to exactly one robot, finding the racks to be visited and the routes of the robots and pickers. The routing decision for a robot (or a picker) is the sequence of storage racks visited by the robot (or the picker). If a robot and a picker arrive at a storage rack that stores an SKU, then the picker picks the number of units of the SKU required by the orders that have been assigned to the robot. If the robot (or the picker) arrives at the storage rack earlier than the picker (or the robot), then the robot (or the picker) should wait at the storage rack for the picker (or the robot) to arrive. In other words, one challenging detail of the problem lies in the possible difference between a robot's arrival time at a storage rack and its start picking time at the rack. The time taken by a picker to pick units at a storage rack for an order is proportional to the number of SKUs that need to be picked for that order. In addition to the above assignment and routing (sequencing) decisions, the timing decision related to the arrival time of robots or pickers at the storage racks is also incorporated into this problem. The objective of this decision problem is to minimize the makespan, that is, the latest time for the robots to complete their assigned orders. Figure 2 illustrates a simple example of a solution, which contains the assignment of four orders to two robots, two robot routes, and two picker routes. In this example, Order 1 (requiring 2 units of SKU A and 1 unit of SKU B) and Order 3 (requiring 1 unit of SKU B, 1 unit of SKU C, and 1 unit of SKU D) are assigned to Robot 1; while, Order 2 (requiring 1 unit of SKU A, 1 unit of SKU B, and 1 unit of SKU C) and Order 4 (requiring 3 units of SKU B) are assigned to Robot 2. For picking the required SKUs for the orders, Robot 1 visits Rack 1, Rack 2, and Rack 3 with the assistance by Picker 1, Picker 1, and Picker 2, respectively; while, Robot 2 visits Rack 3 and Rack 4 with the assistance by Picker 2 and Picker 1, respectively. Figure 2 also illustrates timing decisions for the robots and pickers.

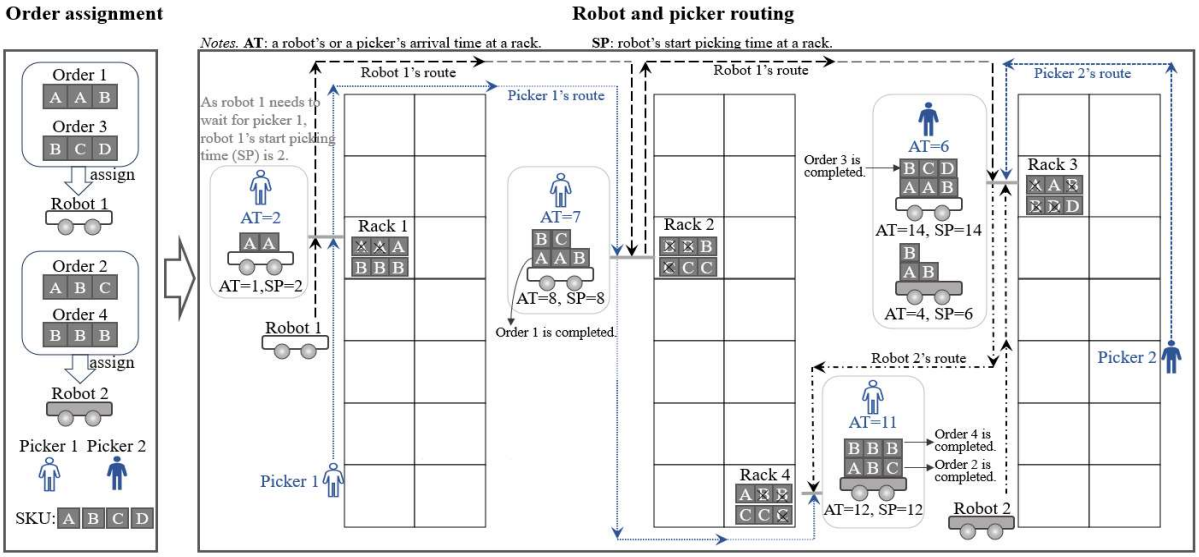

Figure 2: An example of a solution (plan) for the problem

Particularly in e-commerce retail, warehouse operations are driven by random order arrivals, making the environment stochastic. However, this study assumes a deterministic problem, namely, how to handle a batch of known orders. It means that the model must be solved at a certain frequency (e.g. every 3 minutes) to handle order arrivals batch by batch. The orders that arrive within the time period of 3 minutes are considered a batch. For the problem investigated in this study, it is important to set the decision period, which depends mainly on the arrival rate of orders and the time needed to make decisions. This implies that the decisions for the batch of orders accumulated in a period should be made within a duration that is shorter than the period; otherwise, the proposed methodology for the decisions cannot be applied in a real warehouse context with random order arrivals. Effective model formulations and efficient algorithms are needed for this purpose. This study formulates a MILP model for the problem described above and then designs a column-and-row generation algorithm that can solve the model with a batch of orders within a reasonable time. Last but not least, the batches must be integrated in the overall warehouse operational process. In particular, the real-time locations of the pickers and robots and the number of units of SKUs stored in each storage rack must be updated at the end of the execution of a batch. This updated information will be used as the parameters in the decision model to make the plan for the next batch. This leads to a rolling horizon approach based on multiple batches, which can be applied in a warehouse with continuous order arrivals.

Before formulating the model, we state the following assumptions. (1) In a batch of orders, each order is assigned to one robot. We do not consider orders split between multiple robots. However, one robot can be assigned multiple orders. (2) The width of the aisles in the warehouse allows two robots to travel in parallel; thus, the issue of congestion of the robots is not considered in the model. (3) The total quantity of units for each SKU to be picked for all the orders is at least stored in the racks.

4. Mathematical model

This section formulates the MILP model for an integrated optimization problem on assigning orders to robots, selecting storage racks for each order, determining the pick quantity per robot per rack, routing pickers and robots among the selected racks, and sequencing the pickers and the robots that visit each selected rack.

4.1 Notations

Notations for parameters and decision variables used in the model, denoted using Roman and Greek letters, respectively, are listed as follows.

Indices and sets

i index of orders; the set of all the orders is denoted by I

r	index of robots; the set of all the robots is denoted by R
p	index of pickers; the set of all the pickers is denoted by P
s	index of storage racks; the set of all the storage racks is denoted by S
u	index of SKUs; the set of all the SKUs is denoted by U
$o(s), e(s)$	dummy storage racks denoting the start and end points of routes
$o(r), e(r)$	dummy robots denoting the start and end points of storage racks

Parameters

q^I	maximum number of orders assigned to each robot
$n^O_{i,u}$	number of units of SKU u required by order i
$n^S_{s,u}$	number of units of SKU u stored in storage rack s
w^U_u, v^U_u	weight or volume of one unit of SKU u
w^R, v^R	maximum weight or volume of the cargo carried by a robot
$t^R_{s,s\prime}$, $t^P_{s,s\prime}$	travel time for a robot or a picker from storage rack $s \in S$ to rack $s' \in S \cup \{e(s)\}$
$t^{R0}_{r,s}$, $t^{P0}_{p,s}$	travel time for robot r or picker p from its current location to storage rack s
h	unit handling time for picking one SKU from a storage rack and putting it on a robot
M	sufficiently large positive numbers

Remark 1: the big-Ms used by the models in this study can be categorized as six groups of values, which are denoted by $\dot{M}$, $\ddot{M}$, $\ddot{M}^{27}_{s,s'}$, $\ddot{M}^{29}_{s,s'}$, $\ddot{M}^{30}_{s}$ and $\ddot{M}^{31}$. These notations will be used in the different constraints and their values are elaborated in Appendix A.

Decision variables

$\alpha_{i,r}$	equals 1 if order i is assigned to robot r, and otherwise 0
$\beta_{r,s}$	equals 1 if robot r visits storage rack s, and otherwise 0
$\gamma_{p,s}$	equals 1 if picker p visits storage rack s, and otherwise 0
$\vartheta_{r,s,u}$	number of units of SKU u picked from rack s and put on robot r
$\mu_{r,s,u}$	equals 1 if ϑ_{rsu} is a positive number, and otherwise 0
$\delta_{r,s,s'}$	equals 1 if robot r visits storage rack s, then visits storage rack s', and otherwise 0
$\varepsilon_{p,s,s'}$	equals 1 if picker p visits storage rack s, then visits storage rack s', and otherwise 0
$\sigma_{s,r,r'}$	equals 1 if storage rack s is visited by robot r, then by robot r', and otherwise 0
$\tilde{\delta}_{r,s}$	position of storage rack s in robot r's tour, i.e., sequence of racks that the robot visits
$\tilde{\varepsilon}_{p,s}$	position of storage rack s in picker p's tour, i.e., sequence of racks that the picker visits
$\tilde{\sigma}_{s,r}$	position of robot r in storage rack s's sequence of robots that visit the rack
$\epsilon_{r,s}$	robot r's arrival time at storage rack s
$\theta_{p,s}$	picker p's arrival time at storage rack s
$\rho_{r,s}$	robot r's start picking time at storage rack s

4.2 Model formulation

Based on the above definition of the parameters and variables, the MILP model is formulated as follows.

$[\boldsymbol{\mathcal{M}0}]$ Minimize $\max_{r\in R}\{\epsilon_{r,e(s)}\}$ (4-1)

subject to:

$\sum_{r\in R}\alpha_{i,r} = 1$ $\forall i \in I$ (4-2)

$\sum_{i\in I}\alpha_{i,r} \le q^I$ $\forall r \in R$ (4-3)

$\sum_{p\in P}\gamma_{p,s} \le 1$ $\forall s \in S$ (4-4)

$\sum_{p\in P}\gamma_{p,s} \le \sum_{r\in R}\beta_{r,s}$ $\forall s \in S$ (4-5)

$\beta_{r,s} \le \sum_{p\in P}\gamma_{p,s}$ $\forall r \in R, s \in S.$ (4-6)

Objective (4-2) minimizes the makespan, i.e., the latest time that the robots take to complete their assigned tasks. As the proposed model is oriented to one batch of orders in the context of a rolling horizon with multiple batches, the makespan denotes the maximum completion time for all the orders in the batch. In e-commerce retail, the arrival time at the customer depends on the time the batch can be shipped to the parcel carrier's sorting hub, from where orders are sorted by delivery region and route. Hence, choosing the makespan as objective helps to deliver the orders to the customers in a timely fashion.

Constraints (4-2)–(4-6) are related to the assignment. Constraints (4-2) ensure that each order is assigned to one robot. Constraints (4-3) specify the maximum number of orders that can be assigned to each robot. Constraints (4-4) ensure that each storage rack is visited by at most one picker during the order picking process for one batch of orders. This avoids congestion and delays of pickers waiting for each other. We relax the model by deleting this constraint to test the effect. It appears that all optimal solutions for the problem (in instance set ISG1) meet this constraint. Constraints (4-5) guarantee that if a storage rack is not visited by any robot, then it will not be visited by any picker. Constraints (4-6) state that if a storage rack is visited by robots, then it is surely visited by a picker.

We also have constraints related to the picking process.

$\mu_{r,s,u} \le \vartheta_{r,s,u} \le n^S_{s,u}\mu_{r,s,u}$ $\forall r \in R, s \in S, u \in U$ (4-7)

$\vartheta_{r,s,u} \le n^S_{s,u}\beta_{r,s}$ $\forall r \in R, s \in S, u \in U$ (4-8)

$\beta_{r,s} \le \sum_{u\in U}\mu_{r,s,u}$ $\forall r \in R, s \in S$ (4-9)

$\sum_{i\in I} n^O_{i,u}\alpha_{i,r} = \sum_{s\in S}\vartheta_{r,s,u}$ $\forall r \in R, u \in U$ (4-10)

$\sum_{r\in R}\vartheta_{r,s,u} \le n^S_{s,u}$ $\forall s \in S, u \in U$ (4-11)

$\sum_{s\in S}\sum_{u\in U} w^U_u\vartheta_{r,s,u} \le w^R$ $\forall r \in R$ (4-12)

$\sum_{s\in S}\sum_{u\in U} v^U_u\vartheta_{r,s,u} \le v^R$ $\forall r \in R.$ (4-13)

Constraints (4-7)–(4-13) are related to the number of picked units of SKUs. Constraints (4-7) ensure that the number of units of an SKU picked from a storage rack cannot exceed the number of the units of

the SKU stored in the rack. Constraints (4-8) state that if a storage rack is not visited by a robot, the number of units of an SKU picked from the rack by the robot is zero. Constraints (4-9) ensure that if a storage rack is visited by a robot, the number of units of all SKUs picked from the rack by the robot should not be zero. Constraints (4-10) guarantee that the total number of units of an SKU picked by a robot during its trip is equal to the number of units of the SKU required by all the orders that are assigned to the robot in the trip. Constraints (4-11) ensure that the total number of items of an SKU picked from a storage rack cannot exceed the number of items of the SKU stored on the rack. Constraints (4-12) and (4-13) guarantee that the total weight and volume of the cargo carried by a robot in a single trip cannot exceed the respective limits of weight and volume.

We also have the following routing constraints.

$$\sum_{s\in S\cup\{o(s)\}}\delta_{r,s,s'}=\sum_{s\in S\cup\{e(s)\}}\delta_{r,s',s}=\beta_{r,s'} \qquad \forall r\in R, s'\in S \qquad (4\text{-}14)$$

$$\sum_{s\in S\cup\{o(s)\}}\delta_{r,s,e(s)}=\sum_{s\in S\cup\{e(s)\}}\delta_{r,o(s),s}=1 \qquad \forall r\in R \qquad (4\text{-}15)$$

$$\tilde{\delta}_{r,s}+1-\dot{M}\left(1-\delta_{r,s,s'}\right)\le\tilde{\delta}_{r,s'} \qquad \forall r\in R, s\in S\cup\{o(s)\}, s'\in S\cup\{e(s)\} \qquad (4\text{-}16)$$

$$\tilde{\delta}_{r,s}\le\sum_{s'\in S}\beta_{r,s'} \qquad \forall r\in R, s\in S \qquad (4\text{-}17)$$

$$\sum_{s\in S\cup\{o(s)\}}\varepsilon_{p,s,s'}=\sum_{s\in S\cup\{e(s)\}}\varepsilon_{p,s',s}=\gamma_{p,s'} \qquad \forall p\in P, s'\in S \qquad (4\text{-}18)$$

$$\sum_{s\in S\cup\{o(s)\}}\varepsilon_{p,s,e(s)}=\sum_{s\in S\cup\{e(s)\}}\varepsilon_{p,o(s),s}=1 \qquad \forall p\in P \qquad (4\text{-}19)$$

$$\tilde{\varepsilon}_{p,s}+1-\dot{M}\left(1-\varepsilon_{p,s,s'}\right)\le\tilde{\varepsilon}_{p,s'} \qquad \forall p\in P, s\in S\cup\{o(s)\}, s'\in S\cup\{e(s)\} \qquad (4\text{-}20)$$

$$\tilde{\varepsilon}_{p,s}\le\sum_{s'\in S}\gamma_{p,s'} \qquad \forall p\in P, s\in S \qquad (4\text{-}21)$$

$$\sum_{r\in R\cup\{o(r)\}}\sigma_{srr'}=\sum_{r\in R\cup\{e(r)\}}\sigma_{sr'r}=\beta_{r',s} \qquad \forall s\in S, r'\in R \qquad (4\text{-}22)$$

$$\sum_{r\in R\cup\{o(r)\}}\sigma_{s,r,e(r)}=\sum_{r\in R\cup\{e(r)\}}\sigma_{s,o(r),r}=1 \qquad \forall s\in S \qquad (4\text{-}23)$$

$$\tilde{\sigma}_{s,r}+1-\ddot{M}(1-\sigma_{srr'})\le\tilde{\sigma}_{s,r'} \qquad \forall s\in S, r\in R\cup\{o(r)\}, r'\in R\cup\{e(r)\}. \qquad (4\text{-}24)$$

$$\tilde{\sigma}_{s,r}\le\sum_{r'\in R}\beta_{r',s} \qquad \forall r\in R, s\in S. \qquad (4\text{-}25)$$

Constraints (4-14)–(4-22) are the routing constraints. Constraints (4-14)–(4-17) are related to the robot routes. Specifically, Constraints (4-14) ensure that each robot's route is a sequence of storage racks successively visited by the robot. Constraints (4-15) guarantee that each robot's route originates or ends at a dummy vertex. Constraints (4-16) are used to eliminate subtours. Constraints (4-17) determine the upper bound of the variable $\tilde{\delta}_{r,s}$. Constraints (4-18)–(4-21) are related to the picker routes. Constraints (4-22)–(4-25) are related to the sequence of robots that visit each storage rack. The explanations for these two groups of constraints are similar to those of Constraints (4-14)–(4-17).

We also have the following time constraints.

$$\epsilon_{r,s}\ge t_{r,s}^{R0}\delta_{r,o(s),s} \qquad \forall r\in R, s\in S\cup\{e(s)\} \qquad (4\text{-}26)$$

$$\epsilon_{r,s'}\ge\rho_{r,s}+h\sum_{u\in U}\mu_{r,s,u}+t_{s,s'}^{R}-\ddot{M}_{s,s'}^{27}(1-\delta_{r,s,s'})\,\forall r\in R, s\in S, s'\in S\cup\{e(s)\} \qquad (4\text{-}27)$$

$$\theta_{p,s} \geq t^{P0}_{p,s}\varepsilon_{p,o(s),s} \qquad \forall p \in P, s \in S \cup \{e(s)\} \qquad (4\text{-}28)$$

$$\theta_{p,s'} \geq \rho_{r,s} + h\sum_{u\in U}\mu_{r,s,u} + t^{P}_{s,s'} - \ddot{M}^{29}_{s,s'}(1-\varepsilon_{p,s,s'}) \forall r \in R, p \in P, s \in S, s' \in S \cup \{e(s)\} (4\text{-}29)$$

$$\rho_{r,s} \geq \max\{\epsilon_{r,s}, \theta_{p,s}\} - \ddot{M}^{30}_{s}(2-\beta_{r,s}-\gamma_{p,s}) \qquad \forall r \in R, s \in S, p \in P \qquad (4\text{-}30)$$

$$\rho_{r',s} \geq \rho_{r,s} + h\sum_{u\in U}\mu_{r,s,u} - \ddot{M}^{31}(1-\sigma_{srr'}) \qquad \forall r, r' \in R, s \in S \qquad (4\text{-}31)$$

$$\epsilon_{r,s} \leq \ddot{M}^{30}_{s}\beta_{r,s} \qquad \forall r \in R, s \in S \qquad (4\text{-}32)$$

$$\theta_{p,s} \leq \ddot{M}^{30}_{s}\gamma_{p,s} \qquad \forall p \in P, s \in S \qquad (4\text{-}33)$$

$$\rho_{r,s} \leq \ddot{M}^{30}_{s}\beta_{r,s} \qquad \forall r \in R, s \in S. \qquad (4\text{-}34)$$

Constraints (4-26)–(4-34) are time-related constraints. Constraints (4-26) calculate the arrival time of a robot at its first visiting storage rack from its original location. Constraints (4-27) calculate the arrival time of a robot at storage rack s' according to its start picking time, picking duration at previous rack s, and its travel time from rack s to rack s'. Similar to the above two groups, Constraints (4-28) and (4-29) calculate the arrival time of a picker at her/his visiting storage racks. Constraints (4-30) guarantee that the start picking time of a robot at a storage rack is no earlier than the arrival times of the robot and the picker at the rack. Constraints (4-31) ensure that if robot r' arrives at a storage rack later than another robot r, the start picking time of robot r' at the rack is no earlier than the end picking time of robot r at the rack. Constraints (4-32)–(4-34) link the time-related variables with their corresponding binary variables. Specifically, if a robot (or a picker) does not visit a storage rack, the time-related variables for the activities of the robot (or the picker) at the rack are equal to zero.

We also have the following symmetry-breaking constraints.

$$\sum_{s\in S} s\gamma_{p,s} \leq \sum_{s\in S} s\gamma_{p',s} \qquad \forall p, p' \in P, p < p' \qquad (4\text{-}35)$$

$$\sum_{i\in I} i\alpha_{i,r} \leq \sum_{i\in I} i\alpha_{i,r'} \qquad \forall r, r' \in R, r < r'. \qquad (4\text{-}36)$$

There exists a natural symmetry in the problem due to the indistinguishable objects (e.g., pickers and robots). Therefore, Constraints (4-35) and (4-36) are added as symmetry-breaking constraints, which reduce the domain of the solution space. As pickers (and robots) are homogenous in this problem, there exists a symmetry in the plans for pickers (and robots).

We also have the following variable definition constraints.

$$\alpha_{ir}, \beta_{rs}, \gamma_{ps} \in \{0,1\} \qquad \forall i \in I, r \in R, s \in S, p \in P \qquad (4\text{-}37)$$

$$\vartheta_{rsu} \geq 0, \mu_{rsu} \in \{0,1\} \qquad \forall r \in R, s \in S, u \in U \qquad (4\text{-}38)$$

$$\delta_{rss'}, \varepsilon_{pss'} \in \{0,1\} \qquad \forall p \in P, r \in R, s \in S \cup \{o(s)\}, s' \in S \cup \{e(s)\} \qquad (4\text{-}39)$$

$$\epsilon_{rs}, \theta_{ps}, \rho_{rs} \geq 0, \tilde{\delta}_{r,s}, \tilde{\varepsilon}_{p,s} \in Z^{+} \qquad \forall p \in P, r \in R, s \in S \cup \{o(s), e(s)\} \qquad (4\text{-}40)$$

$$\sigma_{sr\ '} \in \{0,1\} \qquad \forall s \in S, r \in R \cup \{o(r)\}, r' \in R \cup \{e(r)\} \qquad (4\text{-}41)$$

$$\tilde{\sigma}_{s,r} \in Z^{+} \qquad \forall s \in S, r \in R \cup \{o(r), e(r)\}. \qquad (4\text{-}42)$$

Constraints (4-37)–(4-42) define the domains of all the decision variables.

The above model contains six parts of intertwined decisions: the assignment of orders to robots, the assignment of storage racks to pickers and robots, determination of the quantity of each SKU to be picked at each rack, robot routing, picker routing, and the sequence of visits by pickers and robots at each storage rack. The problem in this study nests the traditional VRPs as a special case and is thus strongly NP-hard.

Proposition 1: The problem studied in this paper is NP-hard.

Proof: See Appendix B. ■

5. Column-and-row generation solution algorithm

To solve the problem efficiently, this section proposes a column-and-row generation solution algorithm that extends the traditional column generation method by simultaneously generating variables and structural constraints. As discussed in Section 2, the column-and-row generation method has been applied to solve various problems efficiently in recent studies. Here, we use it solve large-scale problem instances with good quality.

5.1 Algorithmic framework

Before introducing the framework of the column-and-row generation solution algorithm, we first analyze the structural features of the proposed MILP model $\mathcal{M}0$. The core of the model is similar to VRP variants to some extent and can be considered a combination of a VRP for pickers and a VRP for robots. As column generation has been utilized to solve VRPs in the literature, we can also apply the Dantzig-Wolfe decomposition to reformulate our problem as an RMP and two PSPs, each of which is oriented to generating columns (plans) for robots or pickers, respectively. The two PSPs are named robot-PSP and picker-PSP and the two types of columns are named robot-columns and picker-columns. In this problem, the robot-columns and the picker-columns are linked with each other due to the intertwined relationships between the plans of a robot and a picker. These relationships are denoted as picker–robot pairs. In addition, a robot's robot-column is also linked with another robot's robot-column; these relationships are denoted as robot–robot pairs. To establish these inherent links, the RMP contains a large number of linking constraints for all possible picker–robot pairs and robot–robot pairs. Moreover, these pairs are further combined with storage racks. Thus, numerous linking constraints must be defined in the RMP. Another special feature is that most of these linking constraints are redundant, because only one picker and a few robots visit one particular storage rack, and only a few racks are visited (selected) out of a number of racks. Numerous redundant linking constraints do not affect the results but may consume a large budget of computing capacity in the solution process. Different from the traditional column generation, column-and-row generation initially builds a short RMP (SRMP) with a few linking constraints. The difference

between the RMP and the SRMP is that the RMP is constructed to contain all constraints from the original problem but with only a subset of all possible variables, whereas the SRMP describes a further restriction of the original problem that contains a subset of all variables and constraints that form the RMP (Muter et al., 2013a; Maher, 2016). In addition to the PSPs that generate columns related to pickers and robots, column-and-row generation also contains a row-generating PSP that generates linking constraints and adds them to the SRMP in iterations, which avoids the redundancy of the numerous linking constraints contained in the initial RMP so as to reduce the solution time. Therefore, this study adopts the methodology of column-and-row generation to solve our proposed model $\mathcal{M}0$. The diagram of our column-and-row generation algorithm is shown in Figure 3.

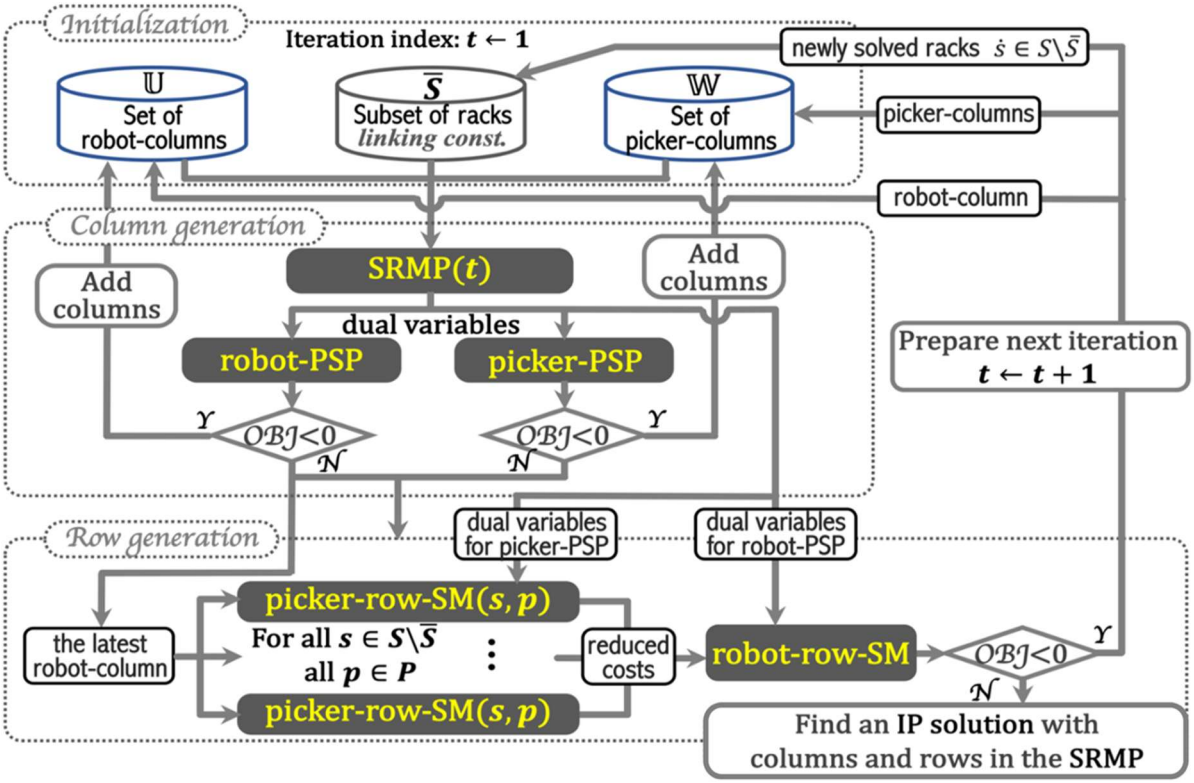

Figure 3: Flow diagram of the column-and-row generation algorithm

The column-and-row generation algorithm first solves the linear programming relaxation of the SRMP; in addition, the very last SRMP is solved by adding integrality restrictions. The column-and-row generation algorithm contains four main components: an SRMP, a robot-PSP, a picker-PSP, and a row-generating PSP. At the beginning of the algorithmic procedure, there are a few linking constraints in the SRMP. As the linking constraints are mainly related to storage racks, we use the set (or subset) of storage racks to represent the linking constraints in this section. Suppose that $\bar{S}$ is the set of storage racks that are related to the linking constraints contained in the SRMP. Based on the dual variables of the SRMP, the robot-PSP and the picker-PSP are constructed and then solved. If the reduced cost of a solved robot-column or a solved picker-column is negative, the columns are added to the SRMP. This process is

repeated iteratively until the reduced costs by solving the robot-PSP and the picker-PSP are no less than zero. For the moment, this would mean that a solution based on the above few linking constraints has been obtained. Next, we have to enter more racks to extend the solution space. This is achieved by extending the current set of linking constraints. Here, the row-generating PSP is used to generate more linking constraints together with robot-columns related to these linking constraints, which are added to the SRMP for the next iteration.

Row generation is an important feature that makes the column-and-row generation algorithm different from a traditional column generation algorithm. The fundamental idea of row generation is to anticipate the dual values of the missing column pairs (or linking constraints) and assess whether they should be introduced to the SRMP so that the objective function value could be improved. As shown in the bottom part of Figure 3 (the part on row generation), our algorithm contains two types of submodels: one robot-row submodel denoted by *robot-row-SM*, and multiple picker-row submodels, each of which is related to a storage rack $s \in S_1$ and denoted by *picker-row-SM(s)*. Here, $S_1 = S \backslash \bar{S}$, which denotes the set of storage racks that are not included in the linking constraints of the SRMP (i.e., "missing column pairs" or linking constraints). In the row generation, the series of picker-row-SMs are constructed on the basis of the dual variables of the picker-PSP; the picker-row-SM's input is the latest robot-column with a negative reduced cost generated by the robot-PSP, and the outputs are a series of reduced costs, each of which is related to a storage rack s. Then, the outputs of the picker-row-SMs (i.e., the reduced costs of the storage racks that are not included in the SRMP) along with the dual variables of the SRMP related to the storage racks (i.e., the reduced costs of the storage racks that are included in the SRMP) constitute the input for the robot-row-SM. The dual variables of the robot-PSP are also used in constructing the robot-row-SM. As shown in the bottom-right corner of Figure 3, "OBJ<0" implies that the purpose of the row generation is to find column pairs that could reduce the objective values once they are introduced into the SRMP.

The last step of the algorithm is to find an IP solution with columns and rows in the SRMP and output the IP solution if the robot-row-SM's objective value is non-negative. Otherwise, the generated robot-column and some of the related picker-columns are added to the SRMP in the next iteration, along with the linking constraints related to the rack (which are not yet included in the SRMP).

The whole procedure of the column-and-row generation algorithm is summarized as follows.

Framework of the column-and-row generation algorithm	
Step 0:	Generate some feasible solutions; based on the solutions, construct set of robot-columns $\mathbb{U}$, set of picker-columns $\mathbb{W}$, and linking constraints; set the iteration index $t \leftarrow 1$, define $\bar{S}$ as the set of storage racks related to these linking constraints, $\bar{S} \subset S$.
Step 1:	Construct and solve **SRMP**(t).
Step 2:	Based on the dual variables of SRMP, construct and solve **robot-PSP** and **picker-PSP**. If the objective value of the robot-PSP's solution (or the picker-PSP's solution) is negative, add the solution, i.e., robot-column (or picker-column), to the set $\mathbb{U}$ (or $\mathbb{W}$), and then go back to Step

1. If both the objective values are non-negative, go to Step 3.

Step 3: Based on the latest robot-column with a negative reduced cost generated by the robot-PSP, construct and solve **picker-row-SM**(s, p) for all $s \in S \backslash \bar{S}$ and $p \in P$.

Step 4 Based on the solutions of picker-row-SMs and the dual variables of the SRMP related to the storage racks, construct and solve **robot-row-SM**; if the objective value is non-negative, go to Step 5; otherwise, the generated robot-column is added to set $\mathbb{U}$, and the linking constraints related to racks $\dot{s} \in S \backslash \bar{S}$, which are in the generated robot-column, are added to the SRMP; the picker-columns, which are generated by the picker-row-SM and whose pickers are related to the newly added racks $\dot{s}$, are added to set $\mathbb{W}$. Let $\bar{S} \leftarrow \bar{S} \cup \{\dot{s}\}$ and $t \leftarrow t + 1$, go to Step 1.

Step 5: Find an IP solution with columns and rows in the SRMP and output the IP solution.

5.2 Preparation before formulating SRMP and other pricing problems

For final algorithmic performance, it is important to formulate a SRMP that can be solved efficiently. Before formulating the SRMP, we need an equivalent transformation for the original model $\mathcal{M}0$. The SRMP as well as other pricing problems are formulated later.

The first challenging issue in formulating the SRMP is that some constraints in model $\mathcal{M}0$ are related to the linking between two robots; it brings much complexity to the SRMP and the pricing problem (i.e., robot-PSP) for generating robot-columns. More specifically, in Constraints (4-22)–(4-25) and (4-31), two robots are linked in each constraint due to the existence of variable $\sigma_{sr'r}$. For circumventing the above issue, we can use the concept, i.e., the position of a robot in the sequence of robots that visit a storage rack, instead of using variable $\sigma_{sr'r}$ to reflect the relative positions of two robots that visit the rack consecutively. Then when the robot-PSP generates a robot-column, we just need to decide a robot's position in a rack's sequence rather than caring whether the robot comes before or after another robot in the rack's sequence. For the above transformation, the following index and variable are defined.

Newly defined index:

k index of a position in a sequence; the set of all the possible positions is denoted by K

Newly defined variable:

$\eta_{s,k,r}$ binary, equal one if robot r is in the k^{th} position of the sequence of the robots that visit storage rack s, otherwise zero

Constraints (4-22)–(4-25) and (4-31) in the $\mathcal{M}0$ can be replaced by Constraints (5-1)–(5-5).

Newly added constraints:

$$\sum_{k \in K} k\eta_{s,k,r} \leq \sum_{r' \in R} \beta_{r',s} \qquad \forall s \in S, r \in R \qquad (5\text{-}1)$$

$$\sum_{r \in R} \eta_{s,k,r} \leq 1 \qquad \forall s \in S, k \in K \qquad (5\text{-}2)$$

$$\sum_{k \in K} \eta_{s,k,r} = \beta_{r,s} \qquad \forall s \in S, r \in R \qquad (5\text{-}3)$$

$$\rho_{r',s} \geq \rho_{r,s} + h \sum_{u \in U} \mu_{r,s,u} - \ddot{M}^{31}\left(2 - \eta_{s,k,r'} - \eta_{s,k-1,r}\right) \quad \forall r, r' \in R, s \in S, k \in K \backslash \{1\} \qquad (5\text{-}4)$$

$$\eta_{s,k,r} \in \{0,1\} \qquad \forall s \in S, k \in K, r \in R. \qquad (5\text{-}5)$$

Constraints (5-1) ensure that the maximum value of a position in a storage rack's sequence should be

no greater than the total amount of robots that visit the rack. Constraints (5-2) state that each position in a sequence should correspond to at most one robot. Constraints (5-3) connect the new variable $\eta_{s,k,r}$ and the binary variable that indicates whether a robot visits a storage rack or not. Constraints (5-4) guarantee that when two robots (i.e., r and r') are in two consecutive positions (i.e., $k-1$ and k), their start picking times are connected. Constraints (5-5) define the domain of the new variable. It is noted that Constraints (5-1), (5-2), and (5-4) are later put in SRMP, while Constraints (5-3) and (5-5) are put in robot-PSP.

The second issue in formulating the SRMP is about Constraints (4-30). If it is put in the SRMP, it is probably infeasible based on the pool of robot-columns generated by the robot-PSP and the pool of the picker-columns generated by the picker-PSP, especially at the beginning stage of the algorithm. It will incur that the number of iterations increases and the solution process becomes time consuming. Therefore, we make the following transformation for the constraints.

Proposition 2: Constraints (4-30) in the $\mathcal{M}0$ can be equivalently replaced by the following two constraints:

$$\rho_{r,s} \geq \epsilon_{r,s} - \ddot{M}_s^{30}\left(1-\beta_{r,s}\right) \qquad \forall r \in R, s \in S \qquad (5\text{-}6)$$

$$\rho_{r,s} \geq \theta_{p,s} - \ddot{M}_s^{30}\left(2-\beta_{r,s}-\gamma_{p,s}\right) \qquad \forall r \in R, s \in S, p \in P. \qquad (5\text{-}7)$$

Proof: See Appendix C. ■

Based on Proposition 2, Constraints (5-6) and (5-7) can be put into the robot-PSP and the SRMP, respectively.

The third issue in formulating the SRMP is about Constraints (4-29). If it is put in the SRMP, it is probably infeasible based on the pool of the picker-columns generated by the picker-PSP. The reason is that Constraints (4-29) have no effect on the generation of picker-columns in the picker-PSP. Thus some "redundant" constraints need to be added to the original model $\mathcal{M}0$. These constraints do not change the optimal solution of the original model, but they can be later put in the picker-PSP so that the generated picker-columns could be feasible for Constraints (4-29), which is later put in the SRMP. Therefore, Proposition 3 is outlined for the above purpose.

Proposition 3: Adding Constraints (5-8) into the $\mathcal{M}0$ does not affect its optimal solution.

$$\theta_{p,s'} \geq \theta_{p,s} + t_{s,s'}^{P} - \ddot{M}_{s,s'}^{29}(1-\varepsilon_{p,s,s'}) \qquad \forall p \in P, s \in S. \qquad (5\text{-}8)$$

Proof: See Appendix D. ■

Based on Proposition 3, Constraints (5-8) can be put into the picker-PSP.

5.3 Short restricted master problem (SRMP)

As shown in Figure 3, the SRMP is a key component in the column-and-row generation. For using the Dantzig & Wolfe (DW) decomposition to reformulate the original model proposed in the previous section, the definition of columns is elaborated first. As aforementioned in Section 5.1, there are two types of columns in this study, i.e., robot-columns and picker-columns. The parameters contained in these two types of columns are also defined as follows.

Indices and sets of columns

$\mathbb{U}$ set of the robot-columns (i.e., plans for robots)

$\mathbb{W}$ set of the picker-columns (i.e., plans for pickers)

Ω_r index of a robot-column for robot r

ϖ_p index of a picker-column for picker p

Parameters for robot-columns

α_{i,Ω_r} equals 1 if order i is assigned to robot r in column (plan) Ω_r, and otherwise 0

β_{s,Ω_r} equals 1 if robot r visits storage rack s in column Ω_r, and otherwise 0

ϑ_{s,u,Ω_r} number of units of SKU u are picked from storage rack s and put on robot r in column Ω_r, and otherwise 0

μ_{s,u,Ω_r} equals 1 if units of SKU u are picked from storage rack s and put on robot r in column Ω_r, and otherwise 0

η_{s,k,Ω_r} equals 1 if robot r is at the k^{th} position in storage rack s's sequence of robots that visit the rack in column Ω_r, and otherwise 0

ρ_{s,Ω_r} robot r's start picking time at storage rack s in column Ω_r

ϵ_{s,Ω_r} robot r's arrival time at storage rack s in column Ω_r

Parameters for picker-columns

γ_{s,ϖ_p} equals 1 if picker p visits storage rack s in column ϖ_p, and otherwise 0

$\varepsilon_{s,s',\varpi_p}$ equals 1 if picker p visits storage rack s, then visits rack s' in column ϖ_p, and otherwise 0

θ_{s,ϖ_p} picker p's arrival time at storage rack s in column ϖ_p

Decision variables for SRMP

λ_{Ω_r} binary, equals 1 if robot-column Ω_r is selected, and otherwise 0

χ_{ϖ_p} binary, equals 1 if picker-column ϖ_p is selected, and otherwise 0

τ makespan, i.e., the latest time of robots' completion of their assigned tasks

Based on the above newly defined columns and variables, the model SRMP is formulated as follows.

[**SRMP**] Minimize τ (5-9)

Subject to:

$$\tau \geq \sum_{\Omega_r \in \mathbb{U}} \epsilon_{\Omega_r, e(s)} \lambda_{\Omega_r} \qquad \forall r \in R \qquad (5\text{-}10)$$

$$\sum_{\Omega_r \in \mathbb{U}} \lambda_{\Omega_r} = 1 \qquad \forall r \in R \tag{5-11}$$

$$\sum_{\varpi_p \in \mathbb{W}} \chi_{\varpi_p} = 1 \qquad \forall p \in P. \tag{5-12}$$

Objective (5-9) is to minimize the makespan by selecting proper robot-columns and picker-columns. Constraints (5-10) determine the makespan. Constraints (5-11) and (5-12) ensure that one robot-column and one picker-column are selected for each robot and each picker, respectively.

$$\sum_{r \in R} \sum_{\Omega_r \in \mathbb{U}} \alpha_{i,\Omega_r} \lambda_{\Omega_r} = 1 \qquad \forall i \in I \tag{5-13}$$

$$\sum_{p \in P} \sum_{\varpi_p \in \mathbb{W}} \gamma_{s,\varpi_p} \chi_{\varpi_p} \leq 1 \qquad \forall s \in S \tag{5-14}$$

$$\sum_{r \in R} \sum_{\Omega_r \in \mathbb{U}} \beta_{s,\Omega_r} \lambda_{\Omega_r} \geq \sum_{p \in P} \sum_{\varpi_p \in \mathbb{W}} \gamma_{s,\varpi_p} \chi_{\varpi_p} \qquad \forall s \in \bar{S} \tag{5-15}$$

$$\sum_{\Omega_r \in \mathbb{U}} \beta_{s,\Omega_r} \lambda_{\Omega_r} \leq \sum_{p \in P} \sum_{\varpi_p \in \mathbb{W}} \gamma_{s,\varpi_p} \chi_{\varpi_p} \qquad \forall s \in \bar{S}, r \in R \tag{5-16}$$

$$\sum_{r \in R} \sum_{\Omega_r \in \mathbb{U}} \vartheta_{s,u,\Omega_r} \lambda_{\Omega_r} \leq n^S_{s,u} \qquad \forall s \in S, u \in U \tag{5-17}$$

$$\sum_{k \in K} \sum_{\Omega_r \in \mathbb{U}} k \eta_{s,k,\Omega_r} \lambda_{\Omega_r} \leq \sum_{r' \in R} \sum_{\Omega_{r'} \in \mathbb{U}} \beta_{s,\Omega_{r'}} \lambda_{\Omega_{r'}} \qquad \forall s \in \bar{S}, r \in R \tag{5-18}$$

$$\sum_{r \in R} \sum_{\Omega_r \in \mathbb{U}} \eta_{s,k,\Omega_r} \lambda_{\Omega_r} \leq 1 \qquad \forall s \in S, k \in K \tag{5-19}$$

$$\sum_{\varpi_p \in \mathbb{W}} \theta_{s',\varpi_p} \chi_{\varpi_p} \geq \sum_{\Omega_r \in \mathbb{U}} \rho_{s,\Omega_r} \lambda_{\Omega_r} + h \sum_{\Omega_r \in \mathbb{U}} \sum_{u \in U} \mu_{s,u,\Omega_r} \lambda_{\Omega_r} + t^P_{s,s'} - \ddot{M}^{29}_{s,s'} (1 - \sum_{\varpi_p \in \mathbb{W}} \varepsilon_{s,s',\varpi_p} \chi_{\varpi_p}) \qquad \forall r \in R, p \in P, s \in \bar{S}, s' \in \bar{S} \cup \{e(s)\} \tag{5-20}$$

$$\sum_{\Omega_r \in \mathbb{U}} \rho_{s,\Omega_r} \lambda_{\Omega_r} \geq \sum_{\varpi_p \in \mathbb{W}} \theta_{s,\varpi_p} \chi_{\varpi_p} - \ddot{M}^{30}_s \left(2 - \sum_{\Omega_r \in \mathbb{U}} \beta_{s,\Omega_r} \lambda_{\Omega_r} - \sum_{\varpi_p \in \mathbb{W}} \gamma_{s,\varpi_p} \chi_{\varpi_p}\right) \qquad \forall r \in R, s \in \bar{S}, p \in P \tag{5-21}$$

$$\sum_{\Omega_{r'} \in \mathbb{U}} \rho_{s,\Omega_{r'}} \lambda_{\Omega_{r'}} \geq \sum_{\Omega_r \in \mathbb{U}} \rho_{s,\Omega_r} \lambda_{\Omega_r} + h \sum_{\Omega_r \in \mathbb{U}} \sum_{u \in U} \mu_{s,u,\Omega_r} \lambda_{\Omega_r} - \ddot{M}^{31} \left(2 - \sum_{\Omega_{r'} \in \mathbb{U}} \eta_{s,k,\Omega_{r'}} \lambda_{\Omega_{r'}} - \sum_{\Omega_{\mathrm{r}} \in \mathbb{U}} \eta_{s,k-1,\Omega_{\mathrm{r}}} \lambda_{\Omega_{\mathrm{r}}}\right) \qquad \forall r, r' \in R, s \in \bar{S}, k \in K \backslash \{1\} \tag{5-22}$$

$$\sum_{s \in S} s \gamma_{s,\varpi_p} \chi_{\varpi_p} \leq \sum_{s \in S} s \gamma_{s,\varpi_{p'}} \chi_{\varpi_{p'}} \qquad \forall p, p' \in P, p < p', \varpi_p \in \mathbb{W} \tag{5-23}$$

$$\sum_{i \in I} i \alpha_{i,r} \lambda_{\Omega_r} \leq \sum_{i \in I} i \alpha_{i,r'} \lambda_{\Omega_{r'}} \qquad \forall r, r' \in R, r < r', \Omega_r \in \mathbb{U}. \tag{5-24}$$

The above Constraints (5-13), (5-14)–(5-16), (5-17), (5-18)–(5-19), (5-20), (5-21), (5-22), (5-23)–(5-24) correspond to Constraints (4-2), (4-4)–(4-6), (4-11), (5-1)–(5-2), (4-29), (5-7), (5-4), (4-35)–(4-36), respectively. Due to the limitation of space, the similar explanations are not repeated here.

$$\lambda_{\Omega_r}, \chi_{\varpi_p} \in \{0,1\} \qquad \forall r \in R, p \in P, \varpi_p \in \mathbb{W}, \Omega_r \in \mathbb{U} \tag{5-25}$$

$$\tau \geq 0. \tag{5-26}$$

Constraints (5-25)–(5-26) define the domains of the decision variables.

It is noted that Constraints (5-15)–(5-16), (5-18), (5-20)–(5-22) are linking constraints. As aforementioned in Section 5.1, $\bar{S}$ is defined as the set of storage racks that are related to the linking constraints contained in the SRMP. Thus the condition "$s \in \bar{S}$" emerges in these constraints. Specifically, set $\bar{S}$ initially contains a few storage racks, which are related to the initial sets of robot-columns and

picker-columns. When the pools of the robot-columns and picker-columns grow gradually during the solution process, set $\bar{S}$ is also being extended in iterations.

After solving the linear programming relaxed SRMP model, a series of dual variables could be obtained and then used as parameters to formulate some other pricing subproblems (PSPs). The definitions of these dual variables are listed in the subsections introducing their corresponding PSPs.

5.4 Pricing subproblem for robots (robot-PSP)

The robot-PSP is used to generate robot-columns in iterations. Before formulating the model of robot-PSP, the newly added parameters and decision variables are defined as follows. The input parameters for the robot-PSP are mainly the values of the dual variables associated with the SRMP's constraints that are related to robots. As the robot-PSP is used to generate robot-columns, the robot-PSP's decision variables should contain the parameters of the robot-columns defined in the previous subsection.

Parameters

π_r^1	dual variable associated with Constraints (5-10), $\pi_r^1 \geq 0$
π_r^{a2}	dual variable associated with Constraints (5-11)
π_i^{c4}	dual variable associated with Constraints (5-13)
π_s^{e6}	dual variable associated with Constraints (5-15), $\pi_s^{e6} \geq 0$
$\pi_{s,r}^{f7}$	dual variable associated with Constraints (5-16), $\pi_{s,r}^{f7} \leq 0$
$\pi_{s,u}^{g8}$	dual variable associated with Constraints (5-17), $\pi_{s,u}^{g8} \leq 0$
$\pi_{s,r}^{j9}$	dual variable associated with Constraints (5-18), $\pi_{s,r}^{j9} \leq 0$
$\pi_{s,k}^{l10}$	dual variable associated with Constraints (5-19), $\pi_{s,k}^{l10} \leq 0$
$\pi_{r,p,s,s'}^{m11}$	dual variable associated with Constraints (5-20), $\pi_{r,p,s,s'}^{m1} \geq 0$
$\pi_{r,s,p}^{n12}$	dual variable associated with Constraints (5-21), $\pi_{r,s,p}^{n12} \geq 0$
$\pi_{r,r',s,k}^{q13}$	dual variable associated with Constraints (5-22), $\pi_{r,r',s,k}^{q13} \geq 0$

Decision variables

α_i	binary, equals 1 if the column contains order i, and otherwise 0
β_s	binary, equals 1 if the column's robot visits storage rack s, and otherwise 0
$\vartheta_{s,u}$	number of units of SKU u picked from storage rack s and put on the column's robot
$\mu_{s,u}$	binary, equals 1 if units of SKU u are picked from storage rack s and put on the column's robot, and otherwise 0
$\delta_{s,s'}$	binary, equals 1 if the column's robot visits storage rack s, then visits rack s', and otherwise 0

$\tilde{\delta}_s$ position of storage rack s in the tour of the column's robot

$\eta_{s,k}$ binary, equals 1 if the column's robot is at the k^{th} position in storage rack s's sequence of robots that visit the rack, and otherwise 0

ρ_s start picking time of the column's robot at storage rack s

ϵ_s arrival time of the column's robot at storage rack s

Based on the above parameters obtained from the SRMP's dual variables and newly defined decision variables, a MILP model is formulated as the robot-PSP.

[**robot-PSP**] Minimize $\pi_r^1\epsilon_{e(s)} - \pi_r^{a2} - \sum_{i\in I}\pi_i^{c4}\alpha_i - \sum_{s\in\bar{S}}\pi_s^{e6}\beta_s - \sum_{s\in\bar{S}}\pi_{s,r}^{f7}\beta_s - \sum_{s\in\bar{S}}\sum_{u\in U}\pi_{s,u}^{g8}\vartheta_{s,u} + \left(\sum_{r'\in R}\sum_{s\in\bar{S}}\pi_{s,r'}^{j9}\beta_s - \sum_{k\in K}\sum_{s\in\bar{S}}\pi_{s,r}^{j9}k\eta_{s,k}\right) - \sum_{k\in K}\sum_{s\in\bar{S}}\pi_{s,k}^{l10}\eta_{s,k} + \sum_{p\in P}\sum_{s\in\bar{S}}\sum_{s'\in\bar{S}\cup\{e(s)\}}\pi_{r,p,s,s'}^{m11}\left(\rho_s + \sum_{u\in U}h\mu_{s,u}\right) - \sum_{p\in P}\sum_{s\in\bar{S}}\pi_{r,s,p}^{n12}\left(\rho_{s,\Omega_r} - \ddot{M}_s^{30}\beta_{s,\Omega_r}\right) + \sum_{r'\in R}\sum_{s\in\bar{S}}\sum_{k\in K\setminus\{1\}}\left\{\pi_{r,r',s,k}^{q13}\left(\ddot{M}^{31}\eta_{s,k-1} + \rho_s + h\sum_{u\in U}\mu_{s,u}\right) + \pi_{r',r,s,k}^{q13}\left(\ddot{M}^{31}\eta_{s,k} - \rho_s\right)\right\}$ (5-27)

Subject to: Constraints (5-3) and (5-6)

$\sum_{i\in I}\alpha_i \le q^I$ (5-28)

$\mu_{s,u} \le \vartheta_{s,u} \le n_{s,u}^S\mu_{s,u}$ $\forall s\in\bar{S}, u\in U$ (5-29)

$\vartheta_{s,u} \le n_{s,u}^S\beta_s$ $\forall s\in\bar{S}, u\in U$ (5-30)

$\sum_{u\in U}\vartheta_{s,u} \ge \beta_s$ $\forall s\in\bar{S}$ (5-31)

$\sum_{i\in I}n_{i,u}^O\alpha_i = \sum_{s\in S}\vartheta_{s,u}$ $\forall u\in U$ (5-32)

$\sum_{s\in\bar{S},}\sum_{u\in U}W_u^U\vartheta_{s,u} \le W^R$ (5-33)

$\sum_{s\in\bar{S},}\sum_{u\in U}V_u^U\vartheta_{s,u} \le V^R$ (5-34)

$\sum_{s'\in\bar{S},\cup\{o(s)\}}\delta_{s',s} = \sum_{s'\in\bar{S},\cup\{e(s)\}}\delta_{s,s'} = \beta_s$ $\forall s\in\bar{S}$ (5-35)

$\sum_{s\in\bar{S}\cup\{o(s)\}}\delta_{s,e(s)} = \sum_{s\in\bar{S}\cup\{e(s)\}}\delta_{o(s),s} = 1$ (5-36)

$\tilde{\delta}_s + 1 - \dot{M}\left(1-\delta_{s,s'}\right) \le \tilde{\delta}_{s'}$ $\forall s\in\bar{S}\cup\{o(s)\}, s'\in\bar{S}\cup\{e(s)\}$ (5-37)

$\tilde{\delta}_s \le \sum_{s'\in S}\beta_{s'}$ $\forall s\in S$ (5-38)

$\epsilon_s \ge t_s^{R0}\delta_{o(s),s}$ $\forall s\in\bar{S}\cup\{e(s)\}$ (5-39)

$\epsilon_{s'} \ge \rho_s + h\sum_{u\in U}\mu_{s,u} + t_{s,s'}^R - \ddot{M}_{s,s'}^{27}(1-\delta_{s,s'})$ $\forall s\in\bar{S}, s'\in\bar{S}\cup\{e(s)\}$ (5-40)

$\epsilon_s \le \ddot{M}_s^{30}\beta_s$ $\forall s\in\bar{S}.$ (5-41)

$\rho_s \le \ddot{M}_s^{30}\beta_s$ $\forall s\in\bar{S}$ (5-42)

Objective (5-27) minimizes the reduced cost of the newly generated robot-column, which is calculated on basis of the values of the dual variables (shadow prices) associated with the SRMP's constraints that are related to robots. The above Constraints (5-28), (5-29)–(5-32), (5-33)–(5-38), (5-39)–(5-40), (5-41), (5-42) correspond to the previous Constraints (4-3), (4-7)–(4-10), (4-12)–(4-17), (4-26)–(4-27), (4-32),

(4-34), respectively. Due to the limitation of space, the similar explanations are not repeated here.

$\alpha_i, \beta_s, \eta_{s,k} \in \{0,1\}$ $\forall i \in I, s \in \bar{S}, k \in K$ (5-43)

$\vartheta_{s,u} \geq 0, \mu_{su} \in \{0,1\}$ $\forall s \in \bar{S}, u \in U$ (5-44)

$\delta_{ss'} \in \{0,1\}, \tilde{\delta}_s \in Z^+$ $\forall s, s' \in \bar{S} \cup \{o(s), e(s)\}$ (5-45)

$\epsilon_s, \rho_s \geq 0$ $\forall s \in \bar{S} \cup \{o(s), e(s)\}.$ (5-46)

Constraints (5-43)–(5-46) define the domains of the decision variables.

5.5 Pricing subproblem for pickers (picker-PSP)

The picker-PSP is used to generate picker-columns in iterations. Before formulating the model of picker-PSP, the newly added parameters and decision variables are defined as follows. The input parameters for the picker-PSP are mainly the values of the dual variables associated with the SRMP's constraints that are related to pickers; some of the picker-PSP's parameters have already been defined in the previous subsection. As the picker-PSP is used to generate picker-columns, the picker-PSP's decision variables should contain the parameters of the picker-columns defined in Section 5.3.

Newly added parameters

π_p^{b3} dual variable associated with Constraints (5-12)

π_s^{d5} dual variable associated with Constraints (5-14), $\pi_s^{d5} \leq 0$

Decision variables

γ_s binary, equals 1 if the column's picker visits storage rack s, and otherwise 0

$\varepsilon_{s,s'}$ binary, equals 1 if the column's picker visits rack s, then visits rack s', and otherwise 0

θ_s arrival time of the column's picker at storage rack s

$\tilde{\varepsilon}_s$ position of storage rack s in the tour of the column's picker

Based on the above parameters obtained from the SRMP's dual variables and newly defined decision variables, a MILP model is formulated as the picker-PSP.

[**picker-PSP**] Minimize $-\pi_p^{b3} - \sum_{s\in\bar{S}} \pi_s^{d5}\gamma_s + \sum_{s\in\bar{S}} \pi_s^{e6}\gamma_s + \sum_{s\in\bar{S}}\sum_{r\in R} \pi_{r,s}^{f7}\gamma_s +$

$\sum_{r\in R}\sum_{s\in\bar{S}} \pi_{r,s,p}^{n12}\left(\theta_s + \dot{M}_s^{30}\gamma_s\right) - \sum_{r\in R}\sum_{s\in\bar{S}}\sum_{s'\in\bar{S}\cup\{e(s)\}} \pi_{r,p,s,s'}^{m11}\left(\theta_{s'} - \ddot{M}_{s,s'}^{29}\varepsilon_{s,s'}\right)$ (5-47)

Subject to: Constraints (5-8)

$\sum_{s'\in\bar{S}\cup\{o(s)\}} \varepsilon_{s',s} = \sum_{s'\in\bar{S}\cup\{e(s)\}} \varepsilon_{s,s'} = \gamma_s$ $\forall s \in \bar{S}$ (5-48)

$\sum_{s\in\bar{S}\cup\{o(s)\}} \varepsilon_{s,e(s)} = \sum_{s\in\bar{S}\cup\{e(s)\}} \varepsilon_{o(s),s} = 1$ (5-49)

$\tilde{\varepsilon}_s + 1 - \dot{M}\left(1 - \varepsilon_{s,s'}\right) \leq \tilde{\varepsilon}_{s'}$ $\forall s \in \bar{S} \cup o(s), s' \in \bar{S} \cup \{e(s)\}$ (5-50)

$\tilde{\varepsilon}_s \leq \sum_{s'\in S} \gamma_{s'}$ $\forall s \in S$ (5-51)

$\theta_s \geq t_s^{P0}\varepsilon_{o(s),s}$ $\forall s \in \bar{S} \cup \{e(s)\}$ (5-52)

$$\theta_s \le \ddot{M}_s^{30}\gamma_s \qquad \forall s \in \bar{S}. \qquad (5\text{-}53)$$

Objective (5-47) minimizes the reduced cost of the newly generated picker-column, which is calculated on basis of the values of the dual variables (shadow prices) associated with the SRMP's constraints that are related to pickers. The above Constraints (5-48)–(5-51), (5-52), (5-53) correspond to Constraints (4-18)–(4-21), (4-28), (4-33), respectively. Due to the limitation of space, the similar explanations are not repeated here.

$$\gamma_s \in \{0,1\} \qquad \forall s \in \bar{S} \qquad (5\text{-}54)$$

$$\theta_s \ge 0, \tilde{\varepsilon}_s \in Z^+ \qquad \forall s \in \bar{S} \cup \{o(s), e(s)\} \qquad (5\text{-}55)$$

$$\varepsilon_{ss'} \in \{0,1\} \qquad \forall s \in S \cup \{o(s)\}, s' \in S \cup \{e(s)\}. \qquad (5\text{-}56)$$

Constraints (5-54)–(5-56) define the domains of the decision variables.

5.6 Submodel for generating rows (robot-row-SM)

The submodel is named as "robot-row-SM" because it not only generates rows but also generates robot-columns. More specifically, when the robot-row-SM is solved for one time, one robot-column is obtained; and one or several storage racks that are outside the set $\bar{S}$ for SRMP and contained in the robot-column could be obtained. Then the linking constraints (i.e., rows) related to the newly obtained storage racks are added to the SRMP; the set $\bar{S}$ also involves the racks. Before formulating the model of robot-row-SM, the newly added parameters and decision variables are defined as follows.

Newly added parameters

$\varphi_{\dot{s},r,p}$ objective value of another submodel (i.e., picker-row-SM); it denotes the minimum reduced cost for adding a robot-column in which robot r visits storage rack $\dot{s}$ where the robot is served by picker p

Besides the above parameter, the dual variables used in the robot-PSP are also the parameters for the robot-row-SM. In addition, the constraints in the robot-PSP are also used in the robot-row-SM, the mathematical model of which is formulated as follows.

[**robot-row-SM**] Minimize $\pi_r^1\epsilon_{e(s)} - \pi_r^{a2} - \sum_{i\in I}\pi_i^{c4}\alpha_i - \sum_{s\in\bar{S}}\pi_s^{e6}\beta_s - \sum_{s\in\bar{S}}\pi_{s,r}^{f7}\beta_s - \sum_{s\in\bar{S}}\sum_{u\in U}\pi_{s,u}^{g8}\vartheta_{s,u} + \left(\sum_{r'\in R}\sum_{s\in\bar{S}}\pi_{s,r'}^{j9}\beta_s - \sum_{k\in K}\sum_{s\in\bar{S}}\pi_{s,r}^{j9}k\eta_{s,k}\right) - \sum_{k\in K}\sum_{s\in\bar{S}}\pi_{s,k}^{l10}\eta_{s,k} + \sum_{p\in P}\sum_{s\in\bar{S}}\sum_{s'\in\bar{S}\cup\{e(s)\}}\pi_{r,p,s,s'}^{m11}\left(\rho_s + \sum_{u\in U}h\mu_{s,u}\right) - \sum_{p\in P}\sum_{s\in\bar{S}}\pi_{r,s,p}^{n12}\left(\rho_{s,\Omega_r} - \ddot{M}^{30}\beta_{s,\Omega_r}\right) + \sum_{r'\in R}\sum_{s\in\bar{S}}\sum_{k\in K\setminus\{1\}}\left\{\pi_{r,r',s,k}^{q13}\left(\ddot{M}^{31}\eta_{s,k-1} + \rho_s + h\sum_{u\in U}\mu_{s,u}\right) + \pi_{r',r,s,k}^{q13}\left(\ddot{M}^{31}\eta_{s,k} - \rho_s\right)\right\} + \sum_{p\in P}\sum_{\dot{s}\in S\setminus\bar{S}}\varphi_{\dot{s},r,p}$ (5-57)

Subject to: Constraints (5-3), (5-6), and (5-28)–(5-42).

Objective (5-57) is the sum of the robot-PSP's objective and the "$\sum_{p\in P}\sum_{\dot{s}\in S\setminus\bar{S}}\varphi_{\dot{s},r,p}$". The former one

is the reduced cost for adding a robot-column under the context with an available set of storage racks $\bar{S}$; the latter one (i.e., $\sum_{p\in P}\sum_{\dot{s}\in S\setminus\bar{S}}\varphi_{\dot{s},r,p}$) is the minimum reduced cost for adding a robot-column under the context with an available set of storage racks $S\setminus\bar{S}$. The sum of them is the minimum reduced cost under the context with the complete set of storage racks S. Objective (5-57) is to minimize the reduced cost and generate an optimal robot-column. The constraints in this model are almost the same as the constraints in the model robot-PSP. It is noted that there is a tiny difference between the above two groups of constraints in the two models. In the constraints of model robot-PSP, it is $s \in \bar{S}$, while it is $s \in S$ in the constraints of the model robot-row-SM.

After solving the robot-row-SM, the newly generated robot-column is added to the SRMP. The linking constraints related to the storage racks, which are not previously included in the SRMP but contained in the above newly generated robot-column, are also added to the SRMP. In addition, some picker-columns, whose pickers are associated with the above newly obtained storage racks, are also added to the SRMP; here the picker-columns are generated by another submodel (i.e., picker-row-SM) that derives parameters $\varphi_{\dot{s},r,p}$ in the objective. The submodel picker-row-SM is elaborated in Appendix E.

5.7 Algorithmic strategies for solving the embedded submodels

The column-and-row generation solution method contains some submodels elaborated in the previous subsections. During the execution of the solution method, these submodels can be solved by CPLEX directly when the problem scale is not extremely large. However, some algorithmic strategies are proposed here for solving submodels directly, in a more efficient way than using CPLEX. We elaborate on strategies for solving the three core submodels: SRMP, robot-PSP, and picker-PSP.

(1) Solving the SRMP

The SRMP is the core of the column-and-row generation solution method. We solve the dual problem of the SRMP (denoted by dual-SRMP) to obtain dual variables for the following steps in the method. The benefit of the following proposition is a considerable decrease in the solution time of the dual-SRMP (also the solution time of the SRMP).

Proposition 4: After the dual variables $\pi^{m1}_{r,p,s,s'}, \pi^{n1}_{r,s,p}, \pi^{q13}_{r,r',s,k}$, and their related parts in constraints are removed, the relaxed dual-SRMP is equivalent to the dual-SRMP.

Proof: See Appendix G. ■

(2) Solving the robot-PSP

For some large-scale instances, the above formulated robot-PSP cannot be solved by CPLEX within a short time, which makes the solution process by the column-and-row generation solution method very time consuming. Therefore, this study proposes a label setting algorithm to solve the robot-PSP efficiently

to accelerate the solution process. The label setting algorithm is a dynamic programming-based algorithm that is usually used to solve the shortest path problem. The details on label definition and label extension function are elaborated in Appendix H.

(3) Solving the picker-PSP

Although the picker-PSP is not as complex as the robot-PSP, this study also designs another label setting algorithm to solve the picker-PSP in a more efficient way than using CPLEX to solve it directly. The details on the label setting algorithm for the picker-PSP are elaborated in Appendix I.

5.8 Method for obtaining a feasible solution

The last step of the algorithm of Figure 3 is to find an IP solution with columns and rows in the SRMP and output it. For some small-scale instances, given the columns (i.e., robot-columns and picker-columns) and the linking constraints that are obtained in the previously elaborated procedure of column and row generation, the SRMP can be solved directly with integral solutions and the whole algorithm terminates. Preliminary experiments show that only for some very small instances (i.e., the ISG1 instances in Table 1), an integral solution of the SRMP can be found directly. Therefore, this subsection proposes a method to construct a near-optimal feasible integral solution based on the robot-columns and picker-columns, which are found in the column and row generation.

The core of the method used to obtain a feasible solution lies in an equivalent model of $\mathcal{M}0$, denoted by $\mathcal{M}0^{Equ}$. The model is formulated on the concepts of three types of templates, i.e., robot-route-templates, picker-route-templates, and rack-SKU-templates, the pools of which are obtained according to the columns (i.e., robot-columns and picker-columns) that are solved in the column and row generation. Here, a robot-route-template (or a picker-route-template) is a sequence of storage racks visited by a robot (or a picker); a rack-robot-SKU-template records the information on which SKUs are picked in a rack when a robot visits it. The details on the model $\mathcal{M}0^{Equ}$ are elaborated in Appendix J.

Note that the column-and-row generation algorithm proposed in Section 5 is not exact, because the method for obtaining an integral solution for the SRMP (proposed above) is a heuristic, rather than an exact method like branch-and-bound. In related work using column generation, the branch-and-bound methodology is usually adopted to obtain an optimal integral solution for the RMP. We used the same idea to make the algorithm exact. However, the branch-and-bound method appears to be much more time-consuming than the heuristic method proposed above and it therefore cannot be applied to realistic warehouse instances; according to some preliminary experiments, the computation time of the branch-and-bound method is about one hundred times the computation time of the heuristic method, while their obtained solutions' objective values are identical. Therefore, this study adopts the method proposed above for obtaining a feasible solution.

5.9 Summary of the column-and-row generation algorithm

Our proposed column-and-row generation algorithm differs from the traditional column generation algorithm in the formulation of the master problem. The master problems in the column-and-row generation and the traditional column generation are denoted by SRMP and RMP, respectively; the letter "S" stands for "small", which implies the restricted master problem in the column-and-row generation has a smaller size than that in the column generation, because not all the linking constraints are initially included in the SRMP. They are gradually generated by the "row generation" and then added into the SRMP. In the RMP, all the linking constraints are included from the beginning. There is a well-known direct relationship between the number of constraints in a problem and the expected time required to solve the linear programming (LP) relaxation. Both types of algorithms include a column generation process, which involves solving the LP relaxation of the RMP and SRMP, where each time a set of columns is added. Since the RMP and SRMP are formulated as very similar problems, with the latter containing fewer constraints, solving the LP for the SRMP requires significantly fewer simplex iterations resulting in faster computation time (Maher, 2016). In Section 6.2, experiments are conducted to compare the column-and-row generation to the column generation in the context of our problem. The results show that column-and-row generation outperforms column generation. It has equal or better solution quality and a (mostly) shorter CPU time. This is because of two reasons. First, our problem involves a large number of linking constraints, which include (i) constraints for linking all possible triples of a robot-column, a picker column, and a rack, and (ii) constraints for linking all possible triples of two robot-columns, and a rack. Second, the majority of these linking constraints are redundant; numerous redundant linking constraints do not affect the results but may consume a large budget of computing capacity in the solution process, which deteriorates the relative performance of the traditional column generation compared to the column-and-row generation in the specific problem context of this study.

6. Numerical experiments

Experiments are conducted to validate the efficiency of the proposed column-and-row generation based solution method; some managerial implications are also derived from sensitivity analysis.

6.1 Experimental settings

The experiments were implemented on a workstation with four Xeon Gold 6154 CPUs (18 cores) running at 3.0 GHz with 512 GB of memory under Windows Server 2012; the models and submodels embedded in the algorithm were implemented with CPLEX 12.6.1 in concert with C#2015. The experiments are based on a warehouse using AMRs implemented by Locus Robotics™ company. The warehouse layout is shown in Figure 4. According to the layout, the speed of the AMRs is 0.7 m/s

(Srinivas and Yu 2022) and the speed of the pickers is 1 m/s (Lee and Murray 2019). The travel time related parameters ($t^R_{s,s\prime}$, $t^P_{s,s\prime}$, $t^{R0}_{r,s}$, $t^{P0}_{p,s}$) are estimated for the problem. As each AMR has six bins as shown in Figure 4, q^I (i.e., the maximum number of orders assigned to each robot) is set to six. We assume that $n^O_{i,u}$ (i.e., the number of units of SKU u required by order i) follows a discrete uniform distribution U[1, 5] and $n^S_{s,u}$ (i.e., the number of units of SKU u stored in a storage rack) follows a discrete uniform distribution U[10, 18]. w^U_u and v^U_u (i.e., the weight and volume of one unit of an SKU) follow discrete uniform distributions U[0.1 kg, 2 kg] and U[0.2 dm^3, 10 dm^3], respectively. w^R and v^R (i.e., the maximum weight or volume of cargo carried by a robot) are set to 36 kg and 100 dm^3, respectively. h (i.e., the unit handling time for picking one SKU from a storage rack and putting it on a robot) is set to 10 seconds.

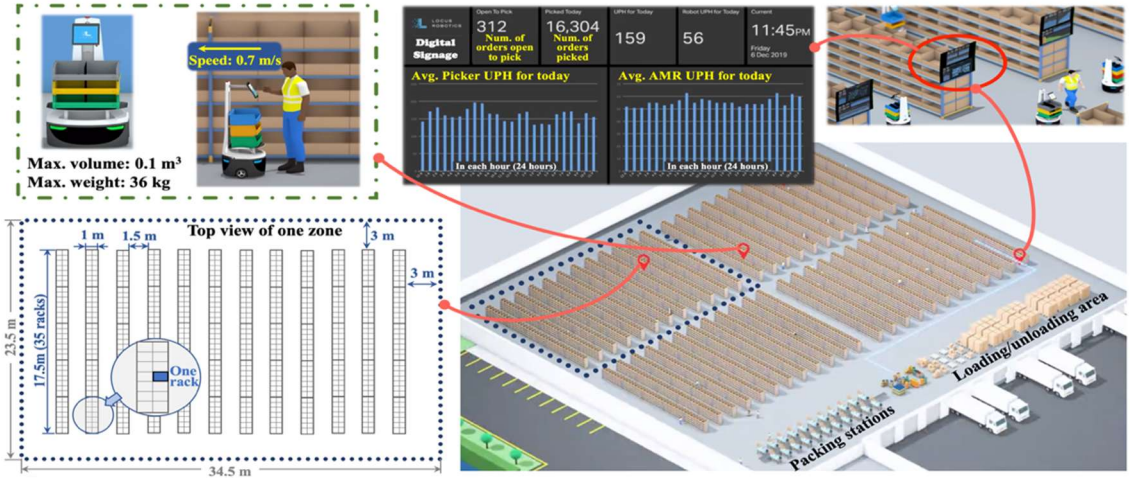

Figure 4: The layout of the warehouse with AMRs

Four instance groups with different scales are generated according to the above-mentioned ranges of the parameters. The instance scale (e.g., ISG 1, …, ISG 4) mainly depends on the size of the key sets, such as the number of orders. Table 1 lists the settings for the four instance scales. Appendix K provides the rationale for setting the values in Table 1. The warehouse handles about 16,000 orders per day. The period for one batch is 3 minutes; Appendix K provides the reason for this. The algorithm should be capable of solving the instances of ISG 4 within 3 minutes; otherwise, the proposed methodology does not apply to the warehouse.

Table 1: Settings for different scales of instance groups (ISGs)

ISG	Num. of orders $\|O\|$	Num. of robots $\|R\|$	Num. of pickers $\|P\|$	Num. of storage racks $\|S\|$	Num. of SKUs $\|U\|$
1	10	2	1	5	8
2	20	4	2	8	14
3	30	5	3	12	21
4	40	8	4	15	27

6.2 Solution quality

The first series of experiments are conducted to validate the quality of solutions solved by our proposed column-and-row generation algorithm. As our algorithm is not exact, we uses the term "solved" to indicate a feasible solution was found. CPLEX software can obtain the optimal result for the instances of ISG1. The gap with the best solution found by CPLEX, denoted by Δ_{CPLEX} in Table 2, is zero for ISG1, which means that the column-and-row generation algorithm can obtain the optimal result in small-scale instances. Our algorithm can also solve the instances of ISG2, which cannot be solved by CPLEX within 2 hours; furthermore, the results solved by our algorithm within a very short time are better than the results obtained by CPLEX taking 2 hours, which is reflected by the negative values in Δ_{CPLEX} for ISG2.

Table 2: Comparison with CPLEX and column generation in small-scale instances

Instances		CPLEX		Column generation		Column-and-row generation			
		OBJ_{CPLEX}	t_{CPLEX}	OBJ_{CG}	t_{CG}	OBJ_{CRG}	t_{CRG}	Δ_{CPLEX}	Δ_{CG}
ISG1	ISG1-1	**62.85**	2.79	**62.85**	4.11	**62.85**	4.06	0%	0%
	ISG1-2	**65.71**	1.28	**65.71**	9.85	**65.71**	2.29	0%	0%
	ISG1-3	**80.00**	2.67	**80.00**	8.41	**80.00**	3.00	0%	0%
	ISG1-4	**75.75**	3.07	85.71	10.81	**75.75**	2.62	0%	-12%
	ISG1-5	**139.00**	3.25	**139.00**	5.26	**139.00**	3.16	0%	0%
	ISG1-6	**75.71**	2.89	**75.71**	5.20	**75.71**	3.13	0%	0%
	ISG1-7	**78.57**	2.39	**78.57**	9.84	**78.57**	2.77	0%	0%
	ISG1-8	**94.28**	1.42	**94.28**	14.43	**94.28**	1.48	0%	0%
ISG2	ISG2-1	166.28	>7200	162.85	21.64	162.83	21.41	-2%	0%
	ISG2-2	158.71	>7200	135.71	25.53	135.71	25.49	-14%	0%
	ISG2-3	150.98	>7200	132.86	15.19	132.86	15.45	-12%	0%
	ISG2-4	150.57	>7200	128.57	13.26	128.57	13.01	-15%	0%
	ISG2-5	138.57	>7200	127.14	28.89	127.14	28.39	-8%	0%
	ISG2-6	151.43	>7200	109.99	37.56	100.00	33.95	-34%	-9%
	ISG2-7	182.71	>7200	145.71	20.82	145.71	27.89	-20%	0%
	ISG2-8	204.71	>7200	142.85	22.37	142.85	16.47	-30%	0%

Notes: $\Delta_{CPLEX}=\frac{(OBJ_{CRG}-OBJ_{CPLEX})}{OBJ_{CPLEX}}$, $\Delta_{CG}=\frac{(OBJ_{CRG}-OBJ_{CG})}{OBJ_{CG}}$. **Bold** numbers are optimal solutions. When Δ_{CPLEX} and Δ_{CG} values are negative, it means the solution obtained by the column-and-row generation algorithm is better than the solution obtained by the CPLEX and the column generation (CG) algorithm, respectively. When "t_{CPLEX}>7200" (for eight instances of ISG2), CPLEX cannot solve the instances to optimality within two hours. A negative Δ_{CPLEX} value for these instances means the solutions found by the column-and-row generation algorithm are better than the best feasible solution obtained by CPLEX within two hours.

We also compare our algorithm with the traditional column generation (CG) method, which is elaborated in Appendix L. The majority of Δ_{CG} values in Table 2 are zero, and one value is negative, which indicates that our algorithm can obtain results that are no worse than (mostly the same as) the results obtained by the CG. In addition, for most instances in Table 2, the solution time of our algorithm is shorter than that of the CG. The reason for the above results is explained in the last paragraph of

Appendix L. We also compare our algorithm and the CG for large-scale instances, and the results are shown in Table 3. The results further validate the relative advantage of our algorithm. Our algorithm's ability to speed up the solving process without sacrificing solution quality becomes more apparent in large-scale experiments.

Table 3: Comparison with the column generation in large-scale instances

Instances		Column generation		Column-and-row generation		
		OBJ_{CG}	t_{CG}	OBJ_{CRG}	t_{CRG}	Δ_{CG}
ISG3	ISG3-1	155.71	178.12	145.71	28.08	-6%
	ISG3-2	305.71	128.69	235.71	36.79	-23%
	ISG3-3	201.43	216.55	201.43	43.43	0%
	ISG3-4	204.85	61.67	204.29	25.57	0%
	ISG3-5	191.42	53.19	191.42	24.80	0%
	ISG3-6	185.71	150.95	182.85	43.09	-2%
	ISG3-7	247.00	110.53	188.57	46.27	-24%
	ISG3-8	224.29	211.58	224.28	47.69	0%
ISG4	ISG4-1	361.71	524.82	347.14	76.42	-4%
	ISG4-2	188.57	324.45	188.57	145.50	0%
	ISG4-3	257.14	646.79	257.14	92.95	0%
	ISG4-4	443.14	834.20	416.42	109.98	-6%
	ISG4-5	234.29	847.35	234.29	150.89	0%
	ISG4-6	322.28	406.79	264.28	111.57	-18%
	ISG4-7	234.28	495.38	234.28	68.79	0%
	ISG4-8	311.42	713.79	251.42	112.53	-19%

Notes: $\Delta_{CG} = \frac{(OBJ_{CRG} - OB\ _{CG})}{OBJ_{CG}}$.

When applying the rolling horizon approach based on multiple batches in the realistic context with continuous order arrivals, the period length may affect the solution quality. We conduct a series of experiments to investigate the influence of different period lengths on the objective value and computation time. The experimental results and analyses are discussed in Appendix M.2. In addition, we also compare our batch-based decision way with the rule-based dynamic decision way; and the results are demonstrated in Table M-2 of Appendix M.3, which shows that the nearest-rule computation time (i.e., decision time) is much shorter than that of batch-based column-and-row generation. However, the nearest-rule results in a much longer makespan than that of batch-based column-and-row generation.

6.3 Algorithmic performance in solving subproblems

In the column-and-row generation, some algorithmic strategies are proposed to solve the submodels. As elaborated in Section 5.7, Proposition 4 is used to accelerate the solving of the SRMP, and two label-setting algorithms are proposed to accelerate the solving of the robot-PSP and the picker-PSP. Experiments are conducted to validate the effectiveness of the proposed algorithmic strategies for solving

the SRMP, the robot-PSP, and the picker-PSP. The detailed results are shown in Table 4 and Table M-1 of Appendix M.

As shown in Table 4, Proposition 4 can accelerate the algorithmic solving process considerably; the computation time is reduced by approximately 71% on average. The results in Table M-1 validate the effectiveness of the proposed label setting algorithms in solving the robot-PSP and picker-PSP. Specifically, the label setting algorithms are compared with CPLEX for solving the PSPs. As CPLEX is very time-consuming for solving them, we set thresholds to the optimality gap and computation time when using CPLEX. The results obtained by CPLEX may not be optimal and are worse than the results obtained by the label setting algorithms, which is reflected by the negative Δ values for some instances in Table M-1. The computation time of the label setting algorithms is shorter than that of CPLEX for all the instances in Table M-1. This validates the effectiveness of the proposed algorithmic strategies embedded in our column-and-row generation solution method.

Table 4: Effect of Proposition 4 in accelerating the solving process

Instances		OBJ_{CRG}	$t_{CRG-SRMP}$	t_{CRG}	$\Delta_{CRG-SRMP}$
ISG3	ISG3-1	145.71	67.81	28.08	-59%
	ISG3-2	235.71	57.76	36.79	-36%
	ISG3-3	201.43	129.56	43.43	-66%
	ISG3-4	204.29	83.38	25.57	-69%
	ISG3-5	191.42	43.88	24.80	-43%
	ISG3-6	182.85	79.08	43.09	-46%
	ISG3-7	188.57	95.60	46.27	-52%
	ISG3-8	224.28	123.63	47.69	-61%
ISG4	ISG4-1	347.14	1865.73	76.42	-96%
	ISG4-2	188.57	632.69	145.50	-77%
	ISG4-3	257.14	1132.19	92.95	-92%
	ISG4-4	416.42	899.99	109.98	-88%
	ISG4-5	234.29	1801.08	150.89	-92%
	ISG4-6	264.28	664.96	111.57	-83%
	ISG4-7	234.28	1795.23	68.79	-96%
	ISG4-8	251.42	679.73	112.53	-83%
					-71%

Notes: (1) OBJ_{CRG} is the objective value of the solution solved by the column-and-row generation algorithm; the computation time is t_{CRG}. (2) When solving the SRMP, we do not use Proposition 4 (elaborated in Section 5.7), the computation time is $t_{CRG-SRMP}$. (3) $\Delta_{CRG-SRMP} = (t_{CRG} - t_{CRG-SRMP})/t_{CRG-SRMP}$.

6.4 Managerial insights

To develop some managerial insights, this section investigates the influence of several key parameters on the final performance (i.e., makespan), based on ISG 4. The two solid curves in Figure 5 show that the makespan decreases as the speed and number of AMRs increase, and their influence on the makespan become less and less significant. The deployment of a large number, or high-speed AMRs entails a high

cost and may also increase potential congestion (and compromise operational safety), while the benefit of decreasing the makespan is limited. The solid curves in Figure 5 imply that eight AMRs with a speed of 1.3 m/s and four pickers might be the appropriate choice for this warehouse system, because when the speed exceeds 1.3 m/s and the number of AMRs exceeds eight, the two curves trend to converge to their lower bounds. The two dashed curves in Figure 5 demonstrate the influence of the AMR loading capacity with respect to the volume and weight of the carried cargo on the makespan. When the capacity limits increase, the makespan improves or remains unchanged (beyond a threshold). In addition, the effect of the loading capacity in weight is more significant on the makespan than the loading capacity in volume.

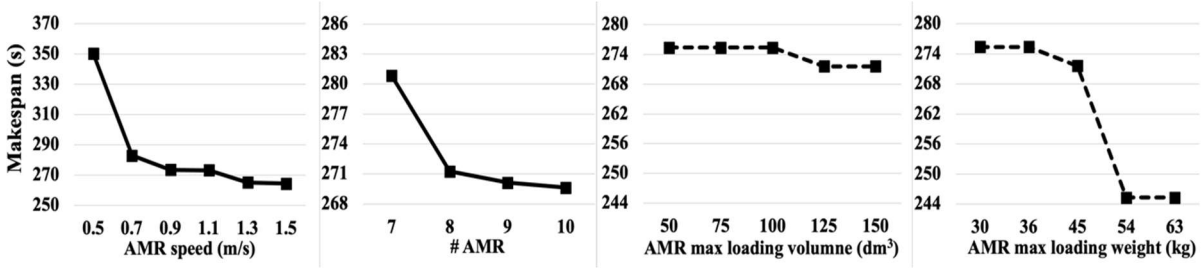

Figure 5: Sensitivity analysis of some of the parameters in the system (ISG 4)

The above sensitivity analyses are based on ISG4, which contains four pickers. It should be noted that the number of pickers affects the makespan given a certain number of AMRs with a certain speed. To investigate this further, we conduct three more experiments, based on the first curve in Figure 5 with four pickers, by varying the number of pickers between three and six. The results in Figure 6 show that with more pickers, the makespan lower bound decreases. However, the 'proper' AMR speed (lowest speed with good performance) is hardly related to the number of pickers. When the number of pickers is three, five, or six, the proper speed of AMRs is about 1.1 m/s. When the number of pickers is four, the proper speed of AMRs is about 1.3 m/s.

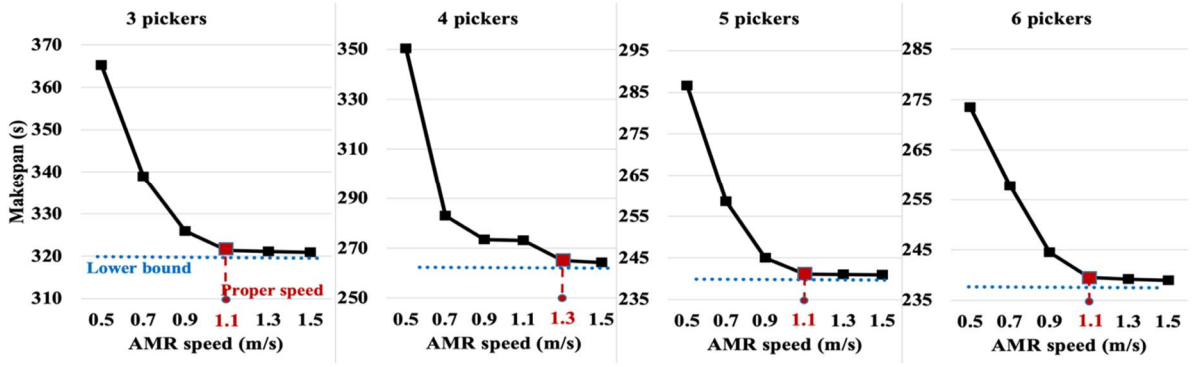

Figure 6: Influence of the number of pickers on the makespan and the proper speed of AMRs

Storage space zoning is an important strategy in warehouse operations management to reduce makespan (see Azadeh et al., 2023). A zoning strategy divides the warehouse into a certain number of disjoint zones (e.g., two or four) and the four pickers are evenly assigned to these zones; each picker is

allowed to move only inside his or her assigned zone. Figure 7 shows the comparative results for different layouts of two or four zones.

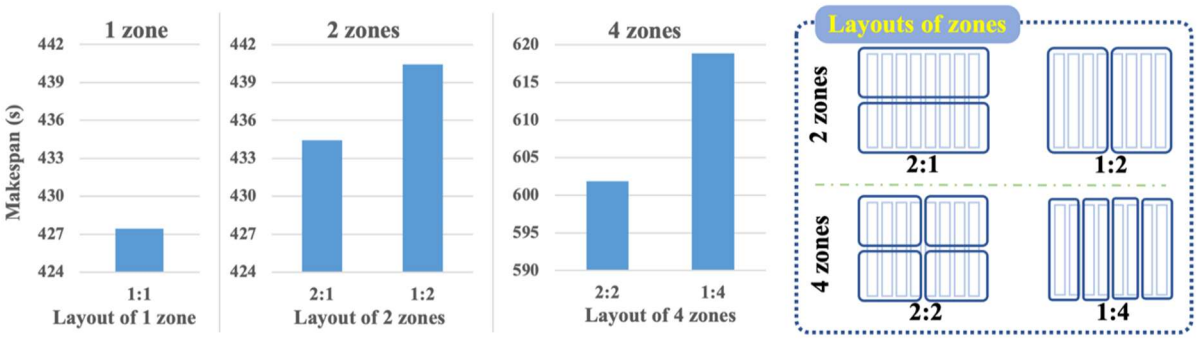

Figure 7: Comparison of different layouts of a certain number of zones

The results in Figure 7 demonstrate the influence of the zone layout on the makespan. For the current order pattern, resulting in routes consisting of about two locations per trip, a single zone appears to be best. In the case of a two-zone layout, the 2:1 layout is better than the 1:2 layout, and in the case of a four-zone layout, the 2:2 layout is better than the 1:4 layout. This is because of the cross-aisles in the middle of the warehouse, which allows making shortcuts. The results in Figure 7 also imply that, for the current order pattern, more zones yield a larger makespan. This aligns with insights by Azadeh et al. (2023) and seems intuitive, because setting fewer zones means a larger solution space for the problem, which improves the solved results.

From the above analyses, some general suggestions can be obtained for warehouse operators. For safety reasons, in many warehouses, AMRs drive at much lower speeds than is possible. However, the makespan is very sensitive to AMR speed. A higher speed significantly reduces the waiting of human pickers for vehicles. It pays off to let the AMRs drive faster, whenever this is possible (that is, there are no humans in the neighborhood) and to increase the number of fast vehicles. However, the additional positive effect of increasing the number of AMRs and their speed gradually decreases. Our mathematical model and solution algorithm can help managers decide on the threshold values for the number of vehicles and their target speed. In addition, the model and algorithm can also support decisions about the zoning strategy and layout when facing the operating context of a specific warehouse. Another interesting finding is that the loading capacity of AMRs also affects their performance. The AMR weight capacity has a more significant influence on the makespan than their volume capacity.

6.5 Robustness of the model against travel time deviations

This study assumes the AMRs' congestion in the warehouse's aisles is not considered, as our numerical example (see Figure 4) is based on a large shelf warehouse, with aisles of 1.5 m wide, using Locus robots; these robots have a footprint of 56×56 cm and can rotate around their vertical axis, so they can easily

overtake each other without delay. However, if we relax this assumption, the AMRs' congestion essentially affects the estimation of the travel time related parameters. More specifically, the travel time between any pair of locations may differ from the estimated value due to unforeseen traffic congestion or other factors in the warehouse operation. Therefore, robustness tests are conducted to investigate whether the performance is robust to the information on the travel time between any pair of two locations (i.e., parameters $t^R_{s,s'}$ and $t^P_{s,s'}$). The results in Figure 8 demonstrate that when the travel times are higher than their estimated values by 5%, 10%, 15%, …, 50%, the order makespan (i.e., OBJ) of the plan solved according to the estimated values is worse than the optimal result using the actual parameters by a percentage gap of less than 0.9%. Thus, the results in Figure 8 validate the robustness of the proposed model against travel time deviations.

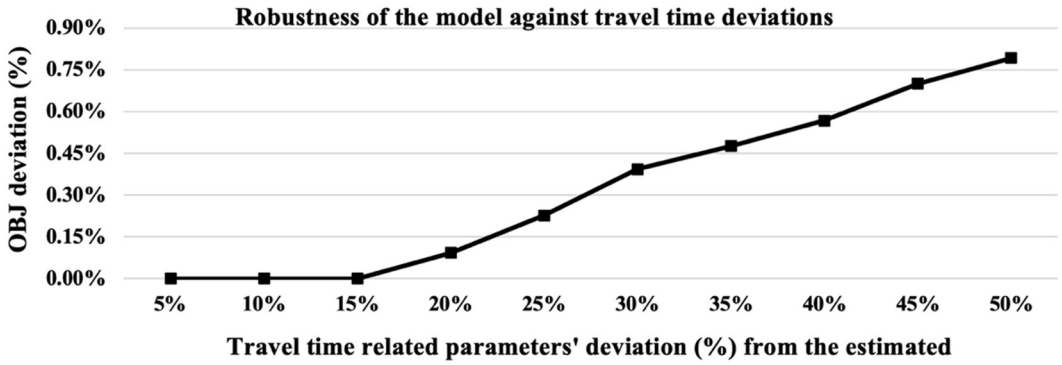

Figure 8: Robustness test of the travel time-related parameters

7. Conclusions

This paper conducts a comprehensive operations optimization study for an AMR-assisted picker-to-parts warehouse system. The model formulation considers several intertwined decisions for the collaborative picking operations of the AMRs and pickers. For solving the model, this study designs a column-and-row generation algorithm that can efficiently handle the synchronization constraints related to the travel and picking activities of the AMRs and the pickers. Numerical experiments are conducted to validate the efficiency of the proposed algorithm and derive some managerial implications for the operators of AMR-assisted warehouses. This study contributes to the literature from the following three perspectives.

From the problem modeling perspective, this study proposes a MILP model to minimize the makespan of handling a batch of orders by determining the order-assignment to AMRs (allowing batches of multiple orders to be picked in one tour), selection of storage racks in a warehouse for AMRs and pickers to visit (assuming scattered storage, taking into account the available quantity of an SKU per rack location), routing paths for AMRs and pickers, and timing their arrival at (and departure from) the storage racks. For each order (or storage rack), the information on its required (or stored) SKUs, and the number of units

for each SKU is incorporated in the model. The model also considers the heterogeneous weight (and volume) for one unit of an SKU and the capacity limit of the AMR with respect to the weight (and volume) of its carried cargo. The proposed model may be the most comprehensive one in the literature on AMR-assisted warehouses. Moreover, this study also contributes to the literature on the VRPSC variants, as our model contains synchronization constraints different from the literature's models. Specifically, we assign orders to AMRs, considering task synchronization between the AMRs and human pickers, and select storage racks from which the orders should be fulfilled with load synchronization, which means the weight and volume of cargos contained in each order, carried by each capacitated robot are considered in picking tasks assignment and scheduling. In addition, both AMRs and pickers must be routed, which requires synchronization at rack positions in space and time, which is integrated into the problem. Furthermore, unlike previous studies, the synchronization positions (i.e., rack positions) are NOT predetermined but are optimized within our proposed model. In conclusion, this problem is novel and has not been dealt with in the VRPSC.

We also contribute with a tailored column-and-row generation algorithm that fits the feature of the problem containing several synchronization constraints between the AMRs and the pickers. As far as we know, our paper is the first to apply the column-and-row generation in a large-scale warehouse order picking optimization problem context; the complexity and scale of our algorithmic application is larger than the related works that have applied the column-and-row generation in some contexts (Muter et al. (2013a; Muter and Sezer, 2018; Yildiz and Savelsbergh, 2019; Wang et al., 2020). In the specific features of the algorithmic design, our algorithm contains two types of columns and pricing problems and two types of submodels for row generation; our algorithmic framework is also more complex than the related algorithms in the literature. This study elaborates the framework and explicit mathematical models on three types of essential subproblems (short RMP, column-generating PSPs, and row-generating submodels), which may provide more detailed guidelines for applying column-and-row generation in more large-scale application contexts. Experiments validate the applicability of our algorithm in a warehouse that handles 16,000 orders per day. Our algorithm can solve much larger scale instances than CPLEX; the solutions gap between the two approaches is zero in small-scale experiments. Our algorithm can also obtain better solutions in less time than the column generation based solution method.

We derive several managerial insights. Deploying more, or faster AMRs may reduce the makespan, but the effect gradually decreases as the speed or the number of AMRs increase. The loading capacity of AMRs also affects the performance; when the capacity increases, the makespan is reduced or remains unchanged (beyond a threshold); and the AMR weight capacity has a more significant influence on the makespan than its volume capacity. In addition, this study derives some insights on area zoning and layout

strategies. For example, more zones yield a larger makespan. Moreover, layouts with a cross-aisle in the middle of the warehouse, which allow making shortcuts, result in a shorter makespan than layouts without the cross-aisles.

The following limitations should be recognized. Our algorithm is not an exact algorithm by nature, although the optimality gap is zero in small-scale experiments. In future research, the column-and-row generation procedure could be embedded inside the branch-and-bound framework to implement an exact algorithm. To this end, the current column-and-row generation procedure should be further accelerated, which will also be beneficial for applying this proposed algorithm to more large-scale realistic problems.

Acknowledgement

This research was supported by the National Natural Science Foundation of China (Grant numbers 72025103, 72394360, 72394362, 72361137001, and 72371221), the Project of Science and Technology Commission of Shanghai Municipality China (grant number 23JC1402200), and the Research Grants Council of the Hong Kong Special Administrative Region, China [Project number HKSAR RGC TRS T32-707/22-N].

References

Alfieri A., Matta A., Pastore E. (2020) The time buffer approximated buffer allocation problem: a row-column generation approach. *Computers & Operations Research* 115: 104835.

Anderluh A., Nolz P. C., Hemmelmayr V. C., Crainic T. G. (2021) Multi-objective optimization of a two-echelon vehicle routing problem with vehicle synchronization and 'grey zone' customers arising in urban logistics. *European Journal of Operational Research* 289(3): 940–958.

Azadeh K., de Koster R., Roy D. (2019) Robotized and automated warehouse systems: review and recent developments. *Transportation Science* 53(4): 917–945.

Azadeh K., Roy D., de Koster R., Khalilabadi S.M.G. (2023) Zoning strategies for human-robot collaborative picking. *Decision Sciences*, https://doi.org.10.1111/deci.12620.

Boysen N., Briskorn D., Emde S. (2017) Parts-to-picker based order processing in a rack-moving mobile robots environment. *European Journal of Operational Research* 262(2): 550–562.

Boysen N., de Koster R., Weidinger F. (2019) Warehousing in the e-commerce era: a survey. *European Journal of Operational Research* 277(2): 396–411.

Chen X., He S., Zhang Y., Tong L., Shang P., Zhou X. (2020) Yard crane and AGV scheduling in automated container port: a multi-robot task allocation framework. *Transportation Research Part C* 114: 241–271.

Drexl M. (2012) Synchronization in vehicle routing: a survey of VRPs with multiple synchronization constraints. *Transportation Science* 46(3): 297–316.

Fink M., Desaulniers G., Frey M., Kiermaier F., Kolisch R., Soumis F. (2019) Column generation for vehicle routing problems with multiple synchronization constraints. *European Journal of Operational Research* 272(2): 699–711.

Fragapane G., de Koster R., Sgarbossa F., Strandhagen J.O. (2021) Planning and control of autonomous mobile robots for intralogistics: literature review and research agenda. *European Journal of Operational Research* 294(2): 405–426.

Gamst M., Lusby R. M., Ropke S. (2024). Exact and heuristic methods for the split delivery vehicle routing problem. *Transportation Science 58*(4): 741–760.

Goeke D., Schneider M. (2021) Modeling single-picker routing problems in classical and modern warehouses. *INFORMS Journal on Computing* 33(2): 436–451.

Hà M., Nguyen T., Nguyen D., Pham H., Do T., Rousseau L. (2020) A new constraint programming model and a linear programming-based adaptive large neighborhood search for the vehicle routing problem with synchronization constraints. *Computers & Operations Research* 124: 105085.

Holte M., Mannino C. (2013) The implementor/adversary algorithm for the cyclic and robust scheduling problem in health-care. *European Journal of Operational Research* 226(3): 551–559.

Lamballais T., Roy D., de Koster M.B.M. (2017) Estimating performance in a robotic mobile fulfillment system. *European Journal of Operational Research* 256(3): 976–990.

Lee H.Y., Murray C.C. (2019) Robotics in order picking: evaluating warehouse layouts for pick, place, and transport vehicle routing systems. *International Journal of Production Research* 57(18): 5821–5841.

Li J., Qin H., Bladacci R., Zhu W. (2020) Branch-and-price-and-cut for the synchronized vehicle routing problem with split delivery, proportional service time and multiple time windows. *Transportation Research Part E* 140: 101955.

Liu R., Tao Y., Xie X. (2019) An adaptive large neighborhood search heuristic for the vehicle routing problem with time windows and synchronized visits. *Computers & Operations Research* 101: 250–262.

Löffler M., Boysen N., Schneider M. (2021) Picker routing in AGV-assisted order picking systems. *INFORMS Journal on Computing* 34(1): 440–462.

Luo Z., Qin H., Zhu W., Lim A. (2016) Branch-and-price-and-cut for the manpower routing problem with synchronization constraints. *Naval Research Logistics* 63(2): 138–171.

Maher S. J. (2016) Solving the integrated airline recovery problem using column-and-row generation. *Transportation Science* 50(1): 216–239.

Meisel F., Kopfer H. (2014) Synchronized routing of active and passive means of transport. *OR Spectrum* 36(2): 297–322.

Muter I., Birbil S. I., Bülbül K. (2013a) Simultaneous column-and-row generation for large-scale linear

programs with column-dependent-rows. *Mathematical Programming* 142(1–2): 47–82.

Muter I., Birbil S. I., Bülbül K., Sahin G., Yenigün H., Tas D., Tüzün D. (2013b) Solving a robust airline crew pairing problem with column generation. *Computers & Operations Research* 40(3): 815–830.

Muter I., Birbil S. I., Bülbül K. (2018) Benders decomposition and column-and-row generation for solving large-scale linear programs with column-dependent-rows. *European Journal of Operational Research* 264(1): 29–45.

Muter I., Sezer Z. (2018) Algorithms for the one-dimensional two-stage cutting stock problem. *European Journal of Operational Research* 271(1): 20–32.

Roy D., Nigam S., de Koster R., Adan I., Resing J. (2019) Robot-storage zone assignment strategies in mobile fulfillment systems. *Transportation Research Part E* 122: 119–142.

Schiffer, M., Boysen, N., Klien, P.S., Laporte, G. (2022) Optimal Picking Policies in E-Commerce Warehouses. *Management Science* 68(10):7497-7517.

Shao S., Xu G., Li M., Huang G. (2019) Synchronizing e-commerce city logistics with sliding time windows. *Transportation Research Part E* 123: 17–28.

Srinivas S., Yu S. (2022) Collaborative order picking with multiple pickers and robots: Integrated approach for order batching, sequencing and picker-robot routing. *International Journal of Production Economics* 254 (2022): 108634.

Suyabatmaz A. Ç., Şahin G. (2015) Railway crew capacity planning problem with connectivity of schedules. *Transportation Research Part E* 84: 88–100.

Tilk C., Bianchessi N., Drexl M., Irnich S., Meisel F. (2018) Branch-and-price-and-cut for the active-passive vehicle routing problem. *Transportation Science* 52(2): 300–319.

Ucar E., Birbil S. I., Muter I. (2017) Managing disruptions in the multi-depot vehicle scheduling problem. *Transportation Research Part B* 105: 249–269.

Wang Z., Sheu J.B., Teo C.P., Xue G. (2022) Robot scheduling for mobile-rack warehouses: human-robot coordinated order picking systems. *Production and Operations Management* 31(1): 98–116.

Wang D., Xiao F., Zhou L., Liang Z. (2020) Two-dimensional skiving and cutting stock problem with setup cost based on column-and-row generation. *European Journal of Operational Research* 286(2): 547–563.

Yildiz B., Savelsbergh M. (2019) Provably high-quality solutions for the meal delivery routing problem. *Transportation Science* 53(5): 1372–1388.

Yu B., Shan W., Sheu J. B., Diabat A. (2022) Branch-and-price for a combined order selection and distribution problem in online community group-buying of perishable products. *Transportation Research Part B: Methodological* 158: 341–373.

Yolmeh A., Baykal-Gürsoy M. (2021) Weighted network search games with multiple hidden objects and multiple search teams. *European Journal of Operational Research* 289(1): 338–349.

Zak E. J. (2002) Row and column generation technique for a multistage cutting stock problem. *Computers & Operations Research* 29(9): 1143–1156.

Zhou H., Qin H., Cheng C., Rousseau L. M. (2023) An exact algorithm for the two-echelon vehicle routing problem with drones. *Transportation research part B: Methodological* 168: 124–150.

Žulj I., Salewski H., Goeke D., Schneider M. (2022) Order batching and batch sequencing in an AMR-assisted picker-to-parts system. *European Journal of Operational Research* 298(1): 182–201.