mathematics

Article

Optimizing Rack Locations in the Mobile-Rack Picking System: A Method of Integrating Rack Heat and Relevance

Mengyue Zhai and Zheng Wang *

School of Maritime Economics and Management, Dalian Maritime University, Dalian 116025, China; zhaimengyue@dlmu.edu.cn
* Correspondence: drwz@dlut.edu.cn

Abstract: The flexible movement of racks in the mobile-rack picking system (MRPS) significantly improves the picking efficiency of e-commerce orders with the characteristics of "one order multi–items" and creates a challenging problem of how to place racks in the warehouse. This is because the placement of each rack in the MRPS directly influences the distance that racks need to be moved during order picking, which in turn affects the order picking efficiency. To handle the rack location optimization problem (RLOP), this work introduces a novel idea and methodology, taking into account the heat degree and the relevance degree of racks, to enhance the efficiency of rack placements in the MRPS. Specifically, a two-stage solution strategy is implemented. In stage 1, an integer programming model (Model 1) is developed to determine the heat and relevance degree of racks, and it can be solved quickly by the Gurobi. Stage 2 entails developing a bi-objective integer programming model (Model 2) with the objective to minimize the travel distances of robots in both heavy load and no-load conditions, using the rack heat and relevance degree as inputs. In light of the challenge of decision coupling and the vast solution space in stage 2, we innovatively propose two lower bounds by slacking off the distance between storage locations. A matheuristic algorithm based on Benders decomposition (MABBD) is designed, which utilizes Benders-related rules to reconstruct Model 2, introduces an enhanced cut and an improved optimal cut with RLOP characteristics, and designs the warm start strategy and the master variable fixed strategy. Given the substantial size of real-life problems, the Memetic algorithm (MA) is specifically devised to address them. Instances of varying sizes are also employed to validate the science and efficacy of the model and algorithm.

Keywords: mobile-rack picking system; rack location optimization; degrees of rack heat and relevance; Benders decomposition; Memetic algorithm

MSC: 90B06

check for updates

Citation: Zhai, M.; Wang, Z. Optimizing Rack Locations in the Mobile-Rack Picking System: A Method of Integrating Rack Heat and Relevance. *Mathematics* **2024**, *12*, 413. https://doi.org/10.3390/math12030413

Academic Editors: Huajun Tang, Jun Wu, Huimin Jiang and Ivan Lai

Received: 31 December 2023
Revised: 23 January 2024
Accepted: 25 January 2024
Published: 26 January 2024

1. Introduction

The rapid development of electronic commerce in recent years has posed substantial implications and challenges for warehouse operation management. Given the demanding timeliness and large-scale, diverse, small-batch nature of customer orders, the urgent resolution of the problem of implementing precise and efficient order picking operations is a critical concern for every warehouse enterprise [1,2]. The MRPS, operating as a semi-automated picking system, provides a fresh way to address this problem [3]. This system deviates from the traditional fixed rack storage system as its racks are smaller in size and can be moved to any desired position within the warehouse. The picking operation is executed by the robot, which carries the rack to the picking station, thereby implementing the "parts to picker" mode. Instead of being stored in a fixed location like a traditional warehouse, each Stock Keeping Unit (SKU) can be scattered and put on numerous mobile racks, each rack holding varying amounts of different categories of SKUs [4,5]. The MRPS significantly improves the flexibility of SKUs storage and efficiently addresses the difficulty of e-commerce order picking by allowing for the relocation of racks in the warehouse [6]. As a result, it is highly desired by numerous e-commerce warehousing enterprises. Presently, e-commerce giants such as JD, Walmart, and Alibaba have embraced MRPS [7].

Nevertheless, this system also gives rise to the challenge of optimizing rack location. The positioning of each rack in the MRPS directly affects the distance traveled by the robots during order picking, thereby impacting the efficiency of the order picking. How can the arrangement of the rack in the warehouse be periodically adjusted, taking into account the regularity of customer orders and the categories and quantities of SKUs stored on the rack, with the objective of minimizing the travel distance of robots? This is a challenging problem that every MRPS decision-maker faces, especially when dealing with e-commerce order picking, in order to maximize efficiency and leverage the system's advantages. This problem is accompanied by the emergence of MRPS, which restricts the development of a new generation of order picking systems. It is different from the location optimization problem in the fixed rack storage system, its solution surpasses the applicability of existing theoretical methods, and its NP-Hard characteristics significantly amplify the problem's difficulty, especially when addressing large-size customer orders [8,9].

In light of this, this paper studies the RLOP in MRPS. A novel approach is proposed to optimize the location of racks by taking into account the categories and quantities of SKUs on each rack, as well as the ordering regularity observed in a large number of customer orders. This method employs the utilization of rack heat and relevance degree and can further enhance the efficacy and scientific rigor of MRPS rack location optimization. The contributions of this paper can be summarized as follows:

(1) A novel idea of "customer orders → racks usage frequency → racks location optimization" is proposed to effectively model and solve the RLOP.
(2) The MABBD algorithm is proposed to solve the integer programming model, and two lower-bound generation methods are designed based on the characteristics of RLOP, which enriches the theoretical research in the field of warehousing optimization.
(3) A Memetic algorithm is specifically developed to address large-scale instances. This method is well-suited to the RLOP, as it employs crossover/mutation operators to rematch racks and locations throughout a broad range, resulting in the creation of a new population. The tabu search operators will explore individual locally by exchanging racks in the same type of locations or racks in the same heat.

The rest of this paper is organized as follows. Section 2 reviews the related work. Section 3 formulates the problem. Section 4 develops the solution algorithm. Section 5 discusses the results of computational experiments. Section 6 concludes the paper and provides future directions.

2. Literature Review

In terms of storage assignment, the traditional manual order picking system solely requires determining the storage position of the SKUs on the rack. This is because the racks have fixed storage locations, and pickers walk in the channels to retrieve SKUs from the racks. In other words, it is a problem of assigning storage locations to SKUs [10]. Hausman et al. were the first to compare three storage assignment strategies: random assignment, full turnover-based assignment, and class-based turnover assignment [11]. Subsequently, Kuo et al. assigned the storage locations with the objective of increasing warehouse utilization and customer satisfaction by reducing customer waiting time [12]. Glock and Grosse studied the problem of storage assignment with the lowest warehouse picking cost [13], while Bortolini et al. optimized SKUs storage locations based on the best rack stability [14]. The objective-setting methods of the above studies are relatively common. In addition, some scholars optimized SKUs storage locations by analyzing certain attributes of the products themselves: Caron et al. proposed a storage strategy based on the Cube-per-Order Index (COI) [15]. Li and Nof proposed a storage optimization method to classify the SKUs based on their attributes [16]. Yang and Nguyen developed a constrained clustering method integrated with principal component analysis to meet the need of clustering stored SKUs with the consideration of practical storage constraints [17]. Since there is a certain correlation between different SKUs in practice and they are often purchased at the same time, other scholars take this correlation into consideration: Pan et al. used the frequency of simultaneous purchases of SKUs to quantify the correlation between products [18]. Xiao and Zheng proposed to classify all SKUs based on correlation and then match them with classified storage locations [19]. Jane and Laih proposed to classify SKUs according to association relationships, and similar SKUs are randomly placed in the same area [20].

The above research focuses on the problem of storage assignment under the traditional "picker-to-parts" model. Within the MRPS system, the robot transports the racks from the storage area to the picking station using the "parts-to-picker" picking mode. Meanwhile, the pickers stand still at the picking station, retrieving the SKUs from the racks to fulfill the orders' picking. After picking is completed, the robot needs to transport the racks from the picking station back to the empty location for storage [21]. Therefore, the storage assignment problem of MRPS can be divided into SKUs storage racks optimization and rack location optimization. Some scholars have used a two-stage solution idea to study the storage assignment problem of MRPS. In the first stage, correlation analysis methods and clustering methods are used to perform correlation clustering on SKUs, so that SKUs that are often purchased at the same time can be stored on the same rack, and to realize that the SKUs stored on racks are maximized in overall relevance. In the second stage, Li et al. proposed a new scattering storage strategy based on rack turnover rate, aiming to decentralize the storage of racks with higher turnover rates to avoid congestion and accumulation of a large number of AGVs in high-frequency operating areas at the same time [22]. Yuan et al. considered rack turnover rate, relevance between racks, and work balance in the lanes to construct an optimization model for minimizing the total travel distance of robots and designed a hybrid algorithm that combines a greedy algorithm and improved simulated annealing [23]. The above research considered both the two sub-problems of SKUs storage optimization and rack location optimization. However, in the real operation of the MRPS, the storage assignment of SKUs on the rack and the adjustment of the rack position are two types of work with different frequencies at different times, so they belong to two different levels of the problem.

Considering that the decision of rack storage location has an important impact on the moving distance of the rack and the workload of the robot, many scholars have conducted research on the RLOP of RMFS in recent years. Weidinger et al. formalized the rack location problem as a special interval scheduling problem by calculating the movement distances between racks and picking stations and introduced a mathematical approach based on adaptive large neighborhood search to solve the problem. It is shown that a simple rack location strategy like the shortest path storage can gain better results when the objective is to minimize the distance between the racks and the picking stations [8]. Yuan et al. developed a fluid model to analyze the performance of velocity-based storage policies. They characterized the maximum possible improvement from applying a velocity-based storage policy in comparison to the random storage policy. The result showed that class-based storage with two or three classes can achieve most of the potential benefits and that these benefits increase with greater variation in the rack velocities [5]. Merschformann et al. introduced several rack repositioning policies such as random policy, fixed policy, and nearest policy [24]. Cai et al. integrated the rack location assignment problem with the path planning decisions to minimize the total travel time of robots [25]. Zhuang et al. investigated the rack storage and robot assignment to racks problem during order processing. They formulated this problem with the objective of minimizing the makespan of this system and developed a matheuristic decomposition approach based on a rolling horizon framework and the simulated annealing method. In the above study, the storage location of the rack is constantly changing, and the rack may be assigned to a new storage location for each completed pick. Merschformann et al. demonstrated that employing variable storage locations is advantageous in numerous scenarios compared to a fixed strategy. This approach effectively minimizes the traveling time of robots, enabling them to promptly accomplish subsequent tasks. However, this strategy may result in no longer useful racks being stored in very prominent storage locations, which is likely to cause problems with congestion and further robot travel distances during order picking later. Moreover, assigning a new storage location after each rack use during the order picking process will greatly increase the computational burden of the intelligent operating system [26,27].

In actual situations, customers have increasingly higher requirements on the delivery time and cost of their orders. Owing to the impact of variables like sales, seasons, product life cycles, and turnover rates, rack locations in MRPS must be changed on a regular basis in order to accommodate these changes and satisfy customer demands. Given that the racks carried to the picking station by the robot always storage numerous SKUs, this paper establishes a correlation between the SKUs combination of customer orders and the SKUs combinations on the racks. By analyzing the SKUs ordering rules disclosed in numbers of customer orders, as well as the categories and quantities of SKUs present on each rack, it is possible to ascertain the using frequency for each rack. Subsequently, the rack locations can be rationally assigned to accommodate future order picking demands. This paper introduces a novel approach to optimize rack location by considering rack heat and relevance. In addition, we also propose a matheuristic algorithm based on Benders decomposition and a memetic algorithm to handle larger-sized real-world cases. This work aims to enhance the scientific rigor and efficacy of rack location optimization in MRPS.

3. Problem Description and Formulation

3.1. Problem Description and Analysis

The picking station in the MRPS is usually fixed in place on one side of the warehouse, as depicted in Figure 1. The position of the rack within the warehouse influences the distance between the rack and the picking station, which in turn, affects the overall travel distance of the robot during picking. The rack's location is regularly updated (every few weeks or months) to accommodate future customer order picking demands, as the categories and quantities of SKUs in the order change. The RLOP in MRPS necessitates the precise arrangement of each rack in the warehouse to reduce the overall distance traveled by the robot when fulfilling customer orders in the future. This paper assumes that orders and their occurrence probabilities are known for the future period (the prediction procedure for order probabilities is not contemplated in this paper). Additionally, it assumes that the categories and quantities of SKUs on each rack are known. Then, the key to optimizing the rack location is determining how to calculate the expected number of times the rack is utilized based on the orders distribution regularity and provide the expected robot travel distance for a given rack location solution.

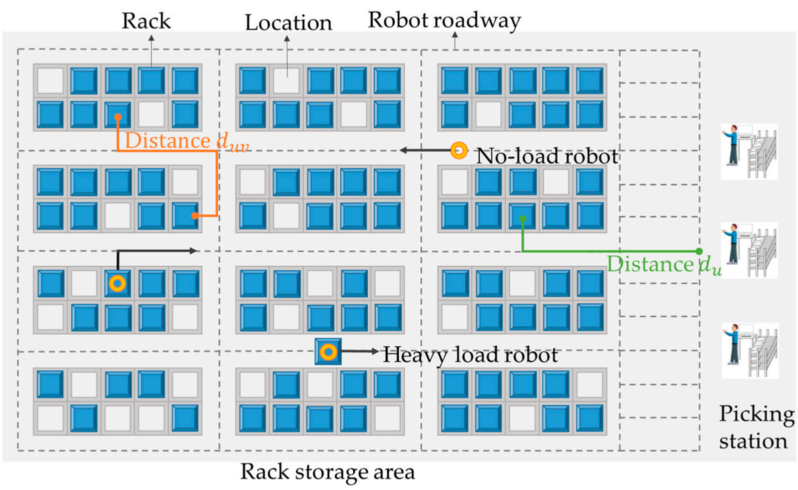

Figure 1. Rack storage location layout in MRPS.

Upon analysis, it is obvious that the robot's movements can be divided into two modes: heavy load, when the robot carries the rack for travel, and no-load, when the robot does the movement itself. The two modes are performed sequentially: during the heavy load mode, the robot carries the rack from its storage location to the picking station, and subsequently returns it from the picking station once the picking process is finished. As soon as the robot completes handling the current rack, it will be no-load onto the next rack and resume the handling operation. The heavy distance traveled by the robot is directly proportional to the expected number of times each rack will be used, taking into account the probability distribution of the orders and the specified rack location arrangement. The no-load distance is determined by the number of times any two racks are carried in succession by the same robot. Considering that various scheduling approaches will result in distinct robot paths, this paper calculates the frequency at which two racks simultaneously serve the same order by utilizing the probability distribution of the orders, in cases when the actual scheduling method utilized by the warehouse is unknown. As the frequency increases, the likelihood of both racks being carried by the same robot consecutively also increases. Therefore, this paper employs such frequencies as a metric to determine the robot no-load travel distance.

Define the expected number of movements of rack i ($\in R$, R is the set of racks) as h_i, denoting the "heat" of the rack. The number of times that two racks i and j are selected to provide picking service for the same order is the "relevance degree" of the two racks, expressed by g_{ij}. The higher the g_{ij}, the greater the possibility that the two racks will be moved by the same robot in turn. The rack location optimization model can be built based on rack heat and relevance degree. Therefore, we first establish the rack heat and correlation generation model to obtain $\{h_i\}_{i\in R}$ and $\{g_{ij}\}_{i,j\in R}$, and then use them as parameters to establish the rack location optimization model.

3.2. Mathematical Model

3.2.1. Rack Heat and Relevance Degree Generation Model (Model 1)

Model 1 aims to determine the appropriate rack to provide picking service for the orders, so as to improve picking efficiency as much as possible. For the same batch of customer orders, there will be a variety of rack selection solutions to meet the picking requirements, but different schemes have different rack movement times. The more times the robot shuttles between the picking station and the rack storage area, the higher the overall distance. Therefore, Model 1 aims to minimize the expected number of rack moves required to complete the picking task and obtain the rack heat and relevance degree attributes. Then, based on these two attributes, the rack location optimization model decides the specific location of each rack with the goal of minimizing the moving distance of the robot.

Let O represent the set of orders to be picked in the future period (several weeks or months), assuming the probability $p_o(0 < p_o \leq 1)$ of order $o(\in O)$ is known (order distribution can be obtained through prediction, and the prediction method is not included in this paper.). All SKUs K required for order picking are stored on racks in set R. The storage capacity of SKU $k(\in K)$ on rack $i(\in R)$ is b_{ik}, and the total supply of items can satisfy the picking demand of orders, and there is no out-of-stock phenomenon. The parameters and decision variables used in Model 1 are defined in Table 1.

Then, Model 1 can be formulated as follows:

$$min\sum_{i\in R}\sum_{o\in O} p_o\cdot x_{io} \tag{1}$$

s.t.

$$y_{iok} \leq d_{ok}\cdot x_{io},\ \forall o \in O, k \in K_o, i \in R_k \tag{2}$$

$$\sum_{i\in R_k} y_{iok} \geq d_{ok},\ \forall o \in O, k \in K_o \tag{3}$$

$$\sum_{k \in K} y_{iok} \leq b_{ik}, \quad \forall o \in O, i \in R \tag{4}$$

$$x_{io} \in \{0, 1\}, \forall i \in R, o \in O \tag{5}$$

$$y_{iok} \in N^{+}, \forall o \in O, k \in K_o, i \in R_k \tag{6}$$

Formula (1) is the objective function that minimizes the expected times of moving racks for picking orders. Constraint (2) formulates the relationship among x and y. It ensures that rack r must not provide items for an order $(y_{iok} = 0)$ when items of an order are not picked from the rack $(x_{io} = 0)$. Constraint (3) ensures that the items of an SKU required by an order must be satisfied by one or more racks. Constraint (4) guarantees that the number of an SKU items picked from a rack must not exceed the number of items stored on the rack. Constraints (5)–(6) define the domains of the variables.

Table 1. Notations used in Model 1.

Parameters					
R	The set of racks.	O	The set of orders.	K	The set of SKUs.
K_o	The set of the SKUs that order $o(\in O)$ contains.				
R_k	The set of racks where SKU $k(\in K)$ is stored.				
b_{ik}	Storage quantity of SKU $k(\in K)$ on rack $i(\in R)$.				
d_{ok}	The number of the items of SKU $k(\in K)$ that order $o(\in O)$ requires.				
p_o	Occurrence probability of order $o(\in O)$.				
g_{ij}	The relevance degree between rack $i(\in R)$ and rack $j(\in R)$, that is, the number of times they provide picking services for the same order.				
h_i	Rack heat, that is, the number of times rack $i(\in R)$ needs to be moved to complete picking task.				
Decision Variables					
x_{io}	A binary variable that equals 1 if order $o(\in O)$ is picked from rack $i(\in R)$.				
y_{iok}	Non-negative integer variable indicating the number of the items of SKU $k(\in K_o)$ picked from rack $i(\in R)$ for order $o(\in O)$.				

3.2.2. Rack Location Optimization Model (Model 2)

On the one hand, rack heat indicates the anticipated number of moves of each rack for future orders; therefore, based on the rack location scheme, we can estimate the travel distance of the robot's heavy load during order picking. The rack relevance degree, conversely, denotes the frequency with which two racks serve the same orders; the greater the frequency, the more likely it is that both racks are sequentially moved by the same robot. Since the number of times two racks are carried sequentially by the same robot depends on the robot scheduling method used in the warehouse, and in the case where the scheduling method is unknown, we cannot accurately know the robot's no-load traveling distance, so this paper calculates the robot's no-load traveling distance based on the rack relevance degree.

It is crucial to highlight that the picking efficiency is influenced to different degrees by the driving speeds of the robot with a heavy load compared to the robot moving with no load (the heavy load has a slower driving speed). Hence, this paper proposes a bi-objective integer programming model to optimize the racking location by minimizing both the heavy load and no-load distances of the robots. The heavy load distance is given higher priority, while the no-load distance is given lower priority. The parameters and decision variables used in Model 2 are shown in Table 2.

Table 2. Notations used in Model 2.

Parameters	
R	The set of racks. L The set of rack locations.
d_u	The distance between the rack location $u(\in L)$ and the picking area.
d_{uv}	The distance between rack location $u(\in L)$ and rack location $v(\in L)$.
g_{ij}	The relevance degree between rack $i(\in R)$ and rack $j(\in R)$, that is, the number of times they provide picking services for the same order.
h_i	Rack heat, that is, the number of times rack $i(\in R)$ needs to be moved to complete picking task.
η_1, η_2	Prioritization or weighting of optimization objectives.
Decision Variables	
m_{iu}	A binary variable that equals 1 if rack $i(\in R)$ is stored in location $u(\in L)$.
n_{ij}	Non-negative variable indicates the distance between rack $i(\in R)$ and rack $j(\in R)$.

Then, Model 2 can be formulated as follows:

$$F = min\eta_1 \sum_{i\in R}\sum_{u\in L} h_i \cdot d_u \cdot m_{iu} + \eta_2 \sum_{i\in R}\sum_{\substack{j\in R\\ j>i}} g_{ij}\cdot n_{ij} \tag{7}$$

s.t.

$$\sum_{u\in L} m_{iu} = 1,\ \forall i \in R \tag{8}$$

$$\sum_{i\in R} m_{iu} \leq 1,\ \forall u \in L \tag{9}$$

$$n_{ij} \geq d_{uv}\cdot\left(m_{iu} + m_{jv} - 1\right),\ \forall i,j \in R, j > i;\ \ u,v \in L, u \neq v \tag{10}$$

$$m_{iu} \in \{0,1\}\ \ \forall i \in R; u \in L \tag{11}$$

$$n_{ij} \geq 0,\ \forall i,j \in R \tag{12}$$

Formula (7) is the objective function that optimizes the weighted sum minimization of the robot's heavy travel distance and no-load travel distance. Constraint (8) ensures that each rack can only be placed in one location. Constraint (9) guarantees that each location can store a maximum of one rack. Constraint (10) formulates the relationship among m and n. Constraints (11)–(12) define the domains of the variables.

4. Solution Algorithm

Given that Model 1 can solve the instance with the size of $|O| = 6000; |R| = 200$ in a short time by Gurobi, for larger instances, a simple batch strategy can be adopted to quickly solve. Hence, Model 1 adeptly addresses the real problem, and the associated intelligent algorithm is not devised in our work. Since the RLOP is a typical unbalanced assignment issue, the complexity of Model 2 is NP-hard. This makes solving realistic large-size problems extremely challenging. When Model 2 encounters an instance with 40 racks and 52 locations, Gurobi achieves a Gap of 81.21% after one hour. Once the number of racks and locations rises to several hundred, solving the model becomes more difficult. Therefore, this section focuses on the optimization method of rack location. Initially, two high-quality lower bounds are derived based on the specific attributes of the RLOP. Subsequently, the matheuristic algorithm based on Benders decomposition (MABBD) is developed to enhance the efficiency of solving Model 2 by taking into account its unique properties. The Memetic algorithm (MA) is specifically developed to address complex problems on a large scale.

4.1. Lower Bound (LB)

In order to improve the performance of the algorithm in this paper, two lower bound acquisition methods are proposed in this section.

The objective function of Formula (7) is relaxed:

$$\begin{aligned} &F = min\eta_1 \sum_{i\in R}\sum_{u\in L} h_i \cdot d_u \cdot m_{iu} + \eta_2 \sum_{i\in R}\sum_{\substack{j\in R\\ j>i}} g_{ij}\cdot n_{ij} \\ &> min\eta_1 \sum_{i\in R}\sum_{u\in L} h_i \cdot d_u \cdot m_{iu} + min\eta_2 \sum_{i\in R}\sum_{\substack{j\in R\\ j>i}} g_{ij}\cdot n_{ij} \\ &= F_1 + F_2 \end{aligned} \quad (13)$$

First, the lower bound F_1^{LB} of F_1 is solved. Select $|R|$ locations closest to the picking station and sort them in ascending order of distance, represented by the set L_1. The racks are sorted by their heat from high to end, represented by the set R_1; Then, a lower bound of F_1 is given by:

$$F_1^{LB} = \eta_1 \sum_{i\in R_1}\sum_{u\in L_1} h_i \cdot d_u \quad (14)$$

Next, the lower bound F_2^{LB} of F_2 is solved, and two methods are designed.

(1) Arrange $g_{ij}(i, j \in R, j > i)$ in ascending order, represented by the set G_1. Similarly, arrange $d_{uv}(u, v \in L, v > u)$ in ascending order by taking the first $|R|(|R| - 1)/2$ values in descending order, represented by the set DR_1. Then, a lower bound of F_2 is given by:

$$F_2^{LB^1} = \eta_2 \sum_{(i,j)\in G_1}\sum_{(u,v)\in DR_1} g_{ij}\cdot d_{uv} \quad (15)$$

(2) For each location $u \in L$, select the nearest $|R| - 1$ locations L_u^2 and calculate the sum of distances from location $u \in L$ to these locations $ds_u = \sum_{v\in L_u^2} d_{uv}$. Arrange $\{ds_u\}_{u\in L}$ in ascending order, the first $|R|$ locations are represented by the set L_2, and their associated distance $\{d_{uv}\}_{u\in L_2, v\in L_u^2}$ in descending order is represented by the set DR_2. Then, the other lower bound of F_2 is given by:

$F_2^{LB} = min\left\{F_2^{LB^1}, F_2^{LB^2}\right\}$. In summary, a lower bound for the original problem can be obtained as:

Theorem 1. *The total travel distance of the robots F is bounded from below by the optimal value of LB.*

4.2. Matheuristic Algorithm Based on Benders Decomposition (MABBD)

Since Model 2 contains two types of decision variables, rack location assignment $\{m_{iu}\}_{i\in R, u\in L}$ and the distance between racks $\{n_{ij}\}_{i\in R, j\in R, j>i}$, Model 2 is reconstructed based on the relevant rules of Benders decomposition. First, the no-load travel distance of the robot in Model 2 is relaxed, and the master problem formed mainly solves the heavy load travel distance of the robot. When the relevant decision variable $\{m_{iu}\}_{i\in R, u\in L}$ is gained, the subproblem solves the no-load travel distance of the robot; that is, it is a linear programming problem and can be solved quickly. In order to address the algorithm's drawback of sluggish convergence, the optimal cut is improved, and the enhanced cut is generated based on the attributes of Model 2. Subsequently, multiple heuristic algorithms are devised to generate compact initial cuts by utilizing warm start strategies that take into account the specific attributes of the RLOP. Ultimately, a solution pool is explicitly built to store approximate optimal solutions. Through the observation of value regularity in these approximate optimal solutions, we identify and choose locations and racks that are highly frequently matched and partially fix their decision variables. Simultaneously, variables that correspond to storage locations and racks that are utilized infrequently are eliminated in order to reduce the search range of the solution space. The algorithm flow is shown in the Figure 2.

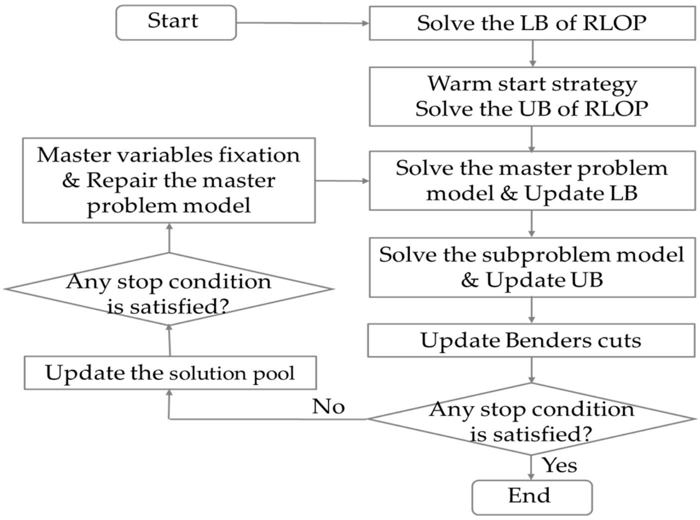

Figure 2. The framework of MABBD.

4.2.1. Master Problem Model

q is defined as the lower bound of the no-load travel distance of the robot in the objective function. The constraints related to the no-load travel of the robot are relaxed to obtain the master problem model as follows:

$$min\ \eta_1 \sum_{i \in R} \sum_{u \in L} h_i \cdot d_u \cdot m_{iu} + \eta_2 \cdot q \tag{16}$$

s.t.

Constraints (10), (11), (13)

$$q \geq F_2^{LB} \tag{17}$$

$$CUTS \tag{18}$$

The objective function (16) minimizes the weighted sum of the robots' heavy-load distance and the lower bounds of the robots' no-load distance. Constraint (17) represents the lower bound of the no-load travel distance on the global search space.

4.2.2. Subproblem Model

When the master problem model is solved, the storage location of each rack $\{m_{iu}\}_{i \in R, u \in L}$ can be determined, and then the subproblem becomes the weighted distance between the racks to minimize the no-load driving distance of the robot. The subproblem model is as follows:

$$q(\boldsymbol{n}) = min \sum_{i \in R} \sum_{\substack{j \in R \\ j > i}} g_{ij} \cdot n_{ij} \tag{19}$$

$$n_{ij} \geq d_{uv} \cdot \left(\overline{m_{iu}} + \overline{m_{jv}} - 1 \right),\ \forall i, j \in R, j > i;\ u, v \in L, u \neq v \tag{20}$$

$$n_{ij} \geq 0,\ \forall i, j \in R, j > i \tag{21}$$

Since the subproblem model must have an optimal solution, it is further simplified as:

$$q(\boldsymbol{n}) = \sum_{i \in R} \sum_{\substack{j \in R \\ j > i}} \sum_{u: \overline{m_{iu}} = 1} \sum_{v: \overline{m_{jv}} = 1} g_{ij} \cdot d_{uv} \tag{22}$$

4.2.3. Benders Cutting

In order to improve the efficiency of the model, this paper further proposes two kinds of Benders cutting.

$$q \geq q\left(\widetilde{\boldsymbol{n}}\right) - \left(q\left(\widetilde{\boldsymbol{n}}\right) - LB_{sp}\right)\left(\sum_{i,u:\widetilde{m}_{iu}=0} m_{iu} + \sum_{i,u:\widetilde{m}_{iu}=1} (1 - m_{iu})\right) \tag{23}$$

$$\begin{cases} \sum\limits_{i,u:\hat{m}_{iu}=0} m_{iu} \geq 1 \\ \sum\limits_{i,u:\hat{m}_{iu}=1} (1 - m_{iu}) \geq 1 \end{cases} \tag{24}$$

When a new upper bound of the original problem is found, cut (23) is generated and added to the master problem. This cut indicates that when the new optimal solution of RLOP is obtained, the no-load distance of robots is at least $q\left(\widetilde{\boldsymbol{n}}\right)$, that is, in the subsequent iteration process, the cut will automatically exclude the current solution, and there is no need to solve it again. When a feasible solution to the original problem is found, cut (24) is generated and added to the master problem. This cut indicates that the current solution is not optimal and need not be considered again. The above valid cuts can convey effective information to the master problem, hence enhancing the solution of the master problem and continuously increasing the lower bound. The process is iterated until either the optimal solution of the RLOP is achieved or the computation time limit is reached.

4.2.4. Warm Start Strategy

According to the characteristics of the RLOP, a variety of heuristic algorithms are designed to tighten the feasible domain, which is relatively relaxed in the initial iteration, and to generate some tight initial cuts and upper bound (UB) of the RLOP.

(1) Bi-direction matching strategy. Firstly, the importance of each rack is calculated $w_i = \eta_1 h_i \overline{d_1} + \eta_2 \overline{g_i}\,\overline{d_2}$, where $\overline{d_1}$ is the average distance from the storage location to the picking station, $\overline{d_2}$ is the average distance between storage locations, and $\overline{g_i}$ is the average relevance degree of rack i. Secondly, the importance of each storage location is calculated as $\omega_u = \frac{1}{\eta_1 d_1 \overline{h} + \eta_2 d_2 \overline{g}}$, where $\overline{h}$ is the average heat, and $\overline{g}$ is the average correlation degree. Finally, the racks and storage locations are matched one by one according to the descending sequence of rack importance and the descending sequence of location importance.

(2) Two-stage solution strategy. In stage 1, the storage locations are divided into ABC areas from near to far according to the distance from the picking station, and the proportion of the number of locations in each area is A = 20%, B = 30%, and C = 50%, respectively. The racks are divided into three areas according to heat. In stage 2, for each area, the rack with the highest heat is selected for priority storage, and then the associated racks of this rack are stored nearby according to the relevance degree.

(3) Integrated matching strategy. The storage location is arranged in ascending sequence of distance from the picking station, and locations that have the same distance from the picking station are considered to be the same location. Racks are allocated for each kind of location in descending sequence of rack heat. When the same type of locations is matched to the same heat of racks, the priority is given to the rack that has a greater relevance degree with racks that have been assigned a location.

(4) Generate feasible solution based on Model 2. The current upper and lower bounds are added to the original model as constraints to solve the linear relaxation solution of the model. The solution is feasible according to the rack importance and location importance of strategy (1).

Through the warm start strategy, four feasible solutions can be obtained, and the feasible solution with the smallest objective value F is taken as the UB of RLOP, and the optimal cut is generated using this feasible solution. Accordingly, the other three feasible solutions generate enhanced cuts.

Theorem 2. *The total travel distance of the robots F is bounded from above by the optimal value of UB.*

4.2.5. Master Variables Fixation Strategy

Considering the warehouse layout, the racks with high heat should be placed as close to the picking station as possible, and the location far away from the picking station is often not the optimal choice, so the value of some decision variables is dominant. This paper introduces a solution pool to store 100 best approximate optimal solutions and record the value rule of each decision variable. When the algorithm is no longer updating the optimal solution in multiple searches, this paper extracts 5% of the decision variables with small changes in their values from the solution pool to fix them, and the model is feasible by checking and deleting the constraints and valid inequalities. Naturally, employing this strategy may discard some of the potential optimal solutions. To effectively prevent this scenario, the strategy permits a maximum usage of six times. In other words, the size of the decision variables we elect to fix must not exceed 30%. Appendix A provides a comprehensive explanation of the justification for the design, utilizing both theoretical analysis and experimentation.

4.3. Memetic Algorithm (MA)

The Memetic algorithm is specifically developed to address the RLOP for large-scale instances. The Memetic algorithm is an optimization technique that combines population intelligence methods with local search methods [28]. Classical intelligent algorithms often have the shortcomings of slow convergence and difficulty finding high-precision solutions when confronted with large-scale and intricate RLOP. Based on these algorithms, the Memetic algorithm introduces a local search strategy to further search the feasible solutions, which will effectively improve the solving efficiency and accuracy of the intelligent algorithm [29,30].

The analysis of RLOP revealed that the choice of rack position has an impact not only on the distance between the rack and the picking station but also on the distance between the racks themselves. Considering the attributes of the warehouse layout, there will be several storage locations equidistant from the picking station, along with several racks that have the same heat. However, the distances between these locations or the relevance degree between the racks may vary. Hence, the crossover/mutation operators of the MA will rematch racks and locations over a wide range to produce a new population. Conversely, the tabu search operators will explore individual locally by exchanging racks in the same type of locations or racks in the same heat. The algorithm flow is shown in Figure 3.

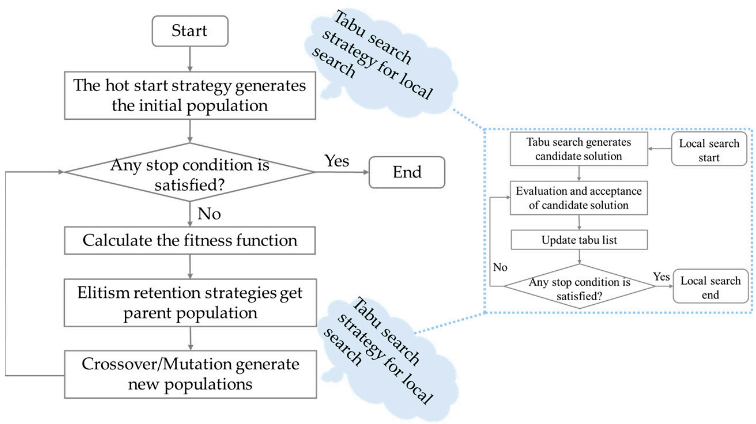

Figure 3. The framework of MA.

4.3.1. Initial Solution Generation

The warm start strategies mentioned in Section 4.2.4 were used to generate multiple initial solutions to form the initial population.

4.3.2. Search Operators

Parameter β_i is introduced to denote the stickiness of rack i with location u, $\{\beta_i\}_{i\in R} = \frac{w_i}{\omega_u}$, where the sticky degree β_i embodies the suitability of matching
$u : m_{iu} = 1$
between a rack and a location. Obviously, the larger the β_i, the better the location assigned to rack i. Let $\overline{\beta} = \sum_{i\in R} \beta_i / |R|$ for the average sticky degree, and racks with a sticky degree lower than $\overline{\beta}$ are filtered, denoted by the set R_{GS}.

(1) Crossover operator

Select $\theta(\theta < |R_{GS}|)$ racks from R_{GS} and use the bi-directional matching strategy in Section 4.2.4 to re-match their locations.

(2) Mutation operator

$\delta(\delta < min\{|R_{GS}|, |L| - |R|\})$ racks are selected from R_{GS}, and δ idle locations are randomly selected, and the bi-directional matching strategy in Section 4.2.4 is used to re-match these locations and racks.

(3) Local search operators

Theorem 3. *In the local search process of the algorithm, the heavy loaded travel distance $F_1$of the robots is unchanged.*

Operator 1: The racks are classified according to heat, and the racks with the same heat are classified as one class. Two racks of the same class are randomly selected to exchange their location.

Operator 2: The locations are classified according to the distance to the picking station, and the locations with the same distance are classified as one class. The two racks stored in the same location are randomly selected for location exchange.

4.3.3. Fitness Function

The fitness function represents the improved value of the optimization objective. Given the unique nature of the RLOP, adjusting the position of each rack only impacts its distance from the picking station and other locations. Therefore, there is no requirement to solve the entire problem again. It is sufficient to calculate the change in travel distance of the racks resulting from the change in location to determine the overall improvement in the problem.

Fitness function of the crossover/mutation operator:

$$\Delta F^{GS} = \eta_1 \sum_{i\in R'} h_i \cdot \left(d^i_{u'} - d^i_u\right) + \eta_2 \sum_{i\in R'} \sum_{j\in R-R'} g_{ij} \cdot \left(d^i_{u'v'} - d^i_{uv}\right) + \eta_2 \sum_{i\in R'} \sum_{\substack{j\in R' \\ j>i}} g_{ij} \cdot \left(d^i_{u'v'} - d^i_{uv}\right) \tag{25}$$

Fitness function of the local search operators:

$$\Delta F^{NS1} = \eta_2 \sum_{i\in R'} \sum_{j\in R-R'} g_{ij} \cdot \left(d_{u'v} - d_{uv}\right) \tag{26}$$

5. Computational Results

To evaluate the effectiveness and performance of the proposed solution approaches, we conducted computational experiments on a computer equipped with an AMD Ryzen7 CPU, 16 GB RAM, and the Windows 10 (64-bit) operating system. All methods presented in this paper are implemented using C#. Gurobi is selected as the solver of mathematical models.

5.1. Instance Generation and Parameter Setting

The rack storage area of the MRPS is composed of multiple blocks; each storage block contains multiple adjacent locations, and the blocks are separated by horizontal and vertical lanes, and the robot can travel in each lane. This paper builds the warehouse layout diagram, as shown in Figure 4, according to the actual layout of the MRPS of a large e-commerce company. The distance of each grid is 1, with a total of 1500 locations. Part of the orders, racks, and SKUs are extracted from the actual business data to form 10 small-scale instances and 20 large-scale instances. The values of relevant parameters are shown in Table 3.

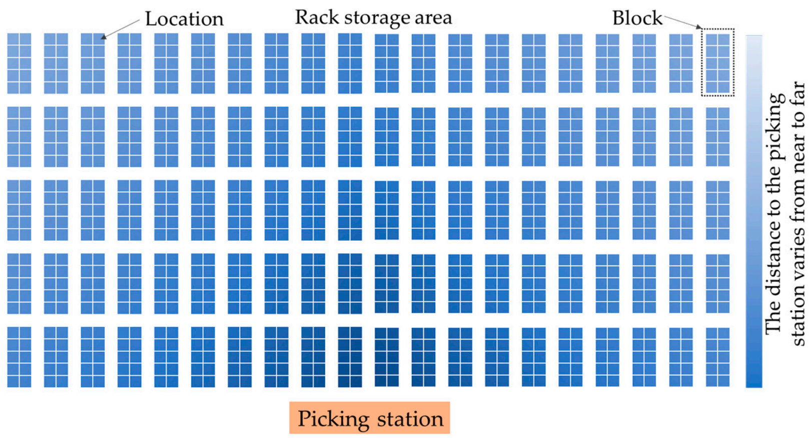

Figure 4. Warehouse layout.

Table 3. The values of relevant parameters.

Algorithm Parameters	Values	RLOP Parameters	Values
Proportion of master variables fixed	0.05	Size of racks	[4, 1359]
Capacity of solution pool	100	Size of locations	[2, 1500]
Population size	50	Distance to picking station	[4, 107]
Elitism retention rate	0.4	Distance between station	[1, 120]
Crossover probability	0.9	Location vacancy rate	[0, 0.3]
Mutation probability	0.1	Size of orders	[15, 30,000]
Global iterations T_G	100	Size of SKUs	[5, 5000]
Local search iterations T_N	$T_n = 100 - T_G$	The categories of SKUs included in an order	[1, 7]
Tabu tenure	10		
The maximum number of un-updated iterations of the global optimal solution.	5	The quantity of each SKU in an order	[1, 5]
The maximum number of un-updated iterations of the local optimal solution.	$min\{10,\ T_n\}$	priority	$\eta_1 = 0.7$ $\eta_2 = 0.3$

5.2. Algorithm Performance Comparison

To evaluate the performance of the MABBD proposed in this paper, this section conducts a comprehensive comparison of the results from our MABBD, MA, Gurobi, and a meta-heuristic (the improved Partheno–Genetical algorithm, IPGA) that has shown good performance for the storage assignment problem. These comparisons are performed across instances of various sizes, with a maximum solution time of 3600 s.

Table 4 displays the results of Gurobi when applied to Model 2, as well as the outcomes of the MABBD, MA, and IPGA proposed in this paper for 10 small and medium-sized instances. In Table 4, Gap_{Gurobi} represents the disparity in results between an algorithm and Gurobi. If the solution time of Gurobi is less than 3600 s, it indicates that Gurobi has found the optimal solution; if the solution time is 3600, it signifies that Gurobi has returned a best-found solution. The solution and running time of other algorithms are the output results after the termination condition of the algorithm is satisfied.

Table 4. Comparison of the solution results on small and medium-sized instances.

No.	Instances $\lvert O\rvert$-$\lvert R\rvert$-$\lvert L\rvert$	$\overline{h}$	$\overline{g}$	Obj		Gap_{Gurobi}			Time (s)			
				Gurobi	**MABBD**	**MABBD**	**MA**	**IPGA**	**Gurobi**	**MABBD**	**MA**	**IPGA**
1	15-4-5	1.8	1.2	85.94	85.94	0.00%	0.00%	0.00%	0.08	0.63	5.06	4.13
2	20-6-8	2.1	1.4	159.95	159.95	0.00%	0.00%	0.00%	1.8	289.3	6.44	4.89
3	28-8-10	1.6	1.1	219.84	219.84	0.00%	0.00%	0.00%	110.17	887.96	8.1	9.7
4	35-10-13	2.1	1.2	254.45	254.45	0.00%	0.00%	0.81%	3600	2150	13.22	13.46
5	76-14-18	2.5	1.1	444.62	432.98	−2.62%	−2.62%	−1.48%	3600	3600	16.5	15.24
6	89-19-25	2.6	0.9	803.93	755.1	−6.07%	−6.07%	−3.56%	3600	3600	22.19	24.78
7	144-25-32	3.3	1.3	1622.37	1546.4	−4.68%	−4.68%	−3.73%	3600	3600	18.8	21.6
8	137-30-39	2.9	1.1	1878.61	1755.16	−6.57%	−6.46%	−5.09%	3600	3600	65.97	88.41
9	166-34-44	3	1.3	2590.65	2309.2	−10.86%	−11.47%	−10.62%	3600	3600	69.84	75.04
10	201-40-52	3.2	1.1	3086.46	2933.5	−4.96%	−5.91%	−3.87%	3600	3600	91.23	112.7
Average	91-19-25	2.51	1.17	1114.68	1045.25	−3.58%	−3.72%	−2.75%	2531.2	2492.7	31.74	37.00

Table 4 shows that when the size of the instances is small, Gurobi can find the optimal solution, but as the size of the instances gradually increases, Gurobi is unable to obtain the optimal solution in 3600s. The performance of the MABBD is better than Gurobi with an average improvement in the objective value of 3.58%. The MA exhibits superior performance, as it not only enhances the objective value by 3.72% but also has a much shorter computation time in comparison to Gurobi. Due to its intelligent nature, the evaluation of the solution's quality cannot be made directly for MA algorithm. While the MABBD can give the upper and lower bound solution of the RLOP (see Appendix B), the average Gap value of the upper and lower bounds obtained by this algorithm is 7.37%, indicating that it has a good convergence impact. Meanwhile, comparing the MDBBD as a benchmark, the MA in this paper shows superior optimization searching ability.

For large-sized instances, due to Gurobi's inability to obtain feasible solutions in a limited period, this paper introduces an additional meta-heuristic algorithm for comparison: the improved Partheno–Genetical algorithm (IPGA). This algorithm is chosen based on its good performance for storage assignment in the recent literature [31–34]. The algorithm details are as follows: drawing on the algorithmic framework of the above literatures, the population size is set to 150, the maximum number of iterations is 1000, and other parameters are set the same as in MA. Let the crossover/mutation operators of MA be the mutation operators of IPGA, and the Tabu search operators of MA be the crossover operators of IPGA.

The results of MA and IPGA on large-sized instances are shown in Table 5. In general, MA has greater advantages than IPGA. The MA results of 20 instances are better than IPGA, saving 60,232.3 units of robot travel distance on average. In order to reflect the performance of MA more intuitively, the results of upper and lower bounds are taken as benchmarks to compare the solving effects of MA and IPGA (see Appendix C for the detailed calculation process and results). Upon analyzing Table 5, it is evident that MA outperforms IPGA in terms of enhancing the objective value, demonstrating an average optimization of 26.40%. Moreover, the optimal objective value found by our MA is closer to the lower bound, which is optimized by 32.54% compared with IPGA. In addition, the running time of the two algorithms is relatively close. Given that RLOP is a strategic decision with a medium to long-term impact, it is not necessary to place excessive emphasis on the algorithm's running time. Our primary focus is on discovering a more optimal resolution.

Table 5. Comparison of the results of large-sized instances.

No.	Instances $\lvert O\rvert$-$\lvert R\rvert$-$\lvert L\rvert$	$\bar{h}$	$\bar{g}$	MA		IPGA		Benchmark	
				Obj	Time (s)	Obj	Time (s)	$\Delta^{UB}_{\mathrm{MA-IPGA}}$	$\Delta^{LB}_{\mathrm{MA-IPGA}}$
11	1176-53-60	19.2	1.1	13,931.9	154.0	14,621.4	110.6	130.61%	−49.25%
12	1687-75-91	22.9	1.3	21,702.0	187.3	22,679.5	172.4	93.94%	−50.65%
13	2346-107-138	21.6	0.9	44,934.2	254.1	46,734.5	211.2	56.97%	−33.73%
14	3335-151-181	20.8	1.1	77,881.8	342.0	80,454.2	366.7	33.30%	−25.75%
15	4689-205-248	17.2	1.2	156,528.5	513.9	163,687.4	499.6	71.59%	−33.93%
16	5613-249-261	20.1	1.4	237,149.0	678.3	240,582.4	617.8	18.49%	−28.49%
17	6467-283-348	25.3	1.0	275,052.2	802.4	282,481.9	833.2	22.71%	−39.17%
18	6802-321-408	22.6	1.1	401,388.3	896.0	414,643.4	918.0	27.91%	−34.69%
19	7992-390-445	20.2	1.3	587,750.8	877.5	607,476.8	922.7	25.19%	−33.38%
20	9537-416-503	24.3	1.2	738,218.8	1132.9	756,478.1	1076.3	23.90%	−29.41%
21	10,585-527-585	19.2	1.0	1,207,114.2	1286.4	1,231,652.0	1325.8	23.04%	−32.47%
22	14,090-616-708	26.6	1.3	1,867,676.3	1481.1	1,901,662.4	1437.2	25.52%	−35.60%
23	15,036-698-768	18.6	0.9	2,402,125.8	1500.7	2,475,922.8	1385.8	31.46%	−32.35%
24	17,687-774-820	20.6	1.3	3,335,183.1	1643.8	3,426,183.2	1593.1	42.09%	−32.63%
25	18,715-855-1060	24.6	1.3	4,020,468.7	1651.4	4,116,579.2	1677.5	24.63%	−33.14%
26	19,461-927-1177	21.3	1.2	4,840,085.3	1589.2	4,921,415.7	1790.2	16.29%	−25.10%
27	23,296-1045-1118	20.6	0.8	6,755,828.4	1854.7	6,867,060.6	1869.4	14.16%	−23.85%
28	25,284-1167-1179	22.3	1.1	8,956,673.4	1965.3	9,028,345.8	1900.3	10.42%	−22.69%
29	27,430-1288-1378	24.4	1.0	11,937,880.1	2032.6	12,185,561.3	2114.7	39.80%	−35.28%
30	29,571-1359-1386	23.5	1.3	13,834,867.1	2418.5	14,132,863.3	2356.2	48.77%	−42.81%
Average	12,540-575-643	21.8	1.1	3,085,622.0	1163.1	3,145,854.3	1158.9	26.40%	−32.54%

In short, the strong solving ability of MA is mainly due to its dual optimization ability of both global search and local search. The crossover/mutation operator disturbs the matching results of racks and locations on a larger range, which is conducive to jumping out of the current neighborhood. The elitism retention strategy also preserves high-quality solutions. The local tabu search strategy further searches for new neighborhood solutions, hence enhancing its ability to identify potential optimal solutions. This cross-application of horizontal search and vertical search is in line with the characteristics of the RLOP, so the optimization ability of the algorithm is improved.

6. Conclusions

Research on the RLOP has significant practical importance in enhancing the efficiency of e-commerce order picking and accelerating the response speed of e-commerce logistics, given the rapid development of the MRPS in e-commerce logistics in recent years. Nevertheless, the key to solving this problem lies in extracting the ordering rules of SKUs in orders and effectively modelling and solving the problem through the process of "customer orders → rack use frequency → rack location optimization". This is particularly important due to the presence of multiple SKUs in e-commerce orders and the practical nature of splitting and storing per SKU on multiple racks in MRPS.

Focusing on the link of "customer orders → rack use frequency", this paper proposes a model that calculates the heat and relevance degree of racks based on customer orders. The objective of the model is to minimize the number of times the racks move. The model also determines the selected picking racks and generates two attribute values: per rack heat and relevance degree between racks. The link "rack frequency → rack position optimization" considers the heavy load and no-load travel distance of the robots as the determining variables for picking efficiency. A bi-objective integer programming model is established to optimize the rack position by minimizing these two distances. To address the challenge of decision coupling and the vast solution space in stage 2, we develop two lower bounds for the original problem by lowering the distance between storage locations. A matheuristic algorithm based on Benders decomposition (MABBD) is designed, which utilizes Benders-related rules to reconstruct Model 2, proposing an enhanced cut and an improved optimal cut, and designing the warm start strategy and the master variable fixed strategy. Given the substantial scale of real-life problems, the Memetic algorithm (MA) is specifically devised to address them. Instances of varying sizes are also employed to validate the science and efficacy of the model and algorithm. The experimental results demonstrate that the suggested algorithm achieves superior quality solutions.

This work introduces a novel idea and methodology for enhancing the efficiency of rack placements in the MRPS. Implementing this approach would significantly enhance the scientific and intelligent aspects of the decision-making process for optimizing rack locations in this system. It would also improve the current inefficient methods utilized for rack location optimization in e-commerce storage management. Our approach efficiently handles extensive customer orders in intricate warehouse operations and improves the applicability and efficiency of optimization theory in warehouse management. According to this study, a crucial and difficult aspect of the MRPS is determining the best location for racks to fulfil customer orders in real-time while also optimizing the robot's scheduling. This will be the focus of future research in this paper.

Author Contributions: Conceptualization, M.Z. and Z.W.; methodology, M.Z.; software, M.Z.; validation, M.Z. and Z.W.; formal analysis, M.Z. and Z.W.; investigation, M.Z.; resources, Z.W.; data curation, Z.W.; writing—original draft preparation, M.Z.; writing—review and editing, M.Z.; visualization, Z.W.; supervision, Z.W.; project administration, Z.W.; funding acquisition, Z.W. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (71971036, 72371047).

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A

This section demonstrates the rationality of the master variables fixation strategy through theoretical analysis and experimental verification.

Considering the characteristics of the RLOP, it appears that the number of locations in the warehouse generally exceeds the number of racks by 0%–30% [8,9]. This indicates that 0–30% of the locations are unnecessary or redundant. Furthermore, racks with a higher heat degree are positioned near the picking station, while racks with a lower heat degree are positioned slightly further away. This arrangement clearly benefits in reducing the robot's travel distance and corresponds to the optimal rack location optimization strategy. Given these factors, it is likely that there are some unnecessary decision variables, which consequently decrease the efficiency of the model's solution. Thus, we anticipate identifying these redundant decision variables using the master variables fixation strategy in order to enhance the algorithm's solution efficiency. It is worth noting that this strategy should not be overused to prevent loss of potential optimal solutions. Therefore, we limited the application of this strategy to fixing only 5% of the decision variables each time it was invoked. Additionally, one of the termination conditions of the MABBD was that the strategy could only be used a maximum of six times. We limit the fixed proportion of decision variables to reduce the probability of losing the optimal solution.

Subsequently, we investigate the reasonableness of this strategy based on empirical data. Given that each instance is computed thrice, the mean of the outcomes is considered as the ultimate result. The frequency of the master variable fixation strategy in each per-instance computation is tallied, and the findings are presented in Table A1. Table A1 indicates that within the effective calculation time, the algorithm invokes the master variable fixation strategy fewer than six times. Combined with Table A2 in Appendix B, it is observed that as the size of the instance increases, the Gap value of the upper and lower bounds obtained by MABBD is not 0. This suggests that we can only determine an interval where the optimal solution exists, and there is still potential for further optimization of either the upper or lower bounds. Nevertheless, our upper bound surpasses the objective solution obtained by Gurobi within the effective calculation time. The utilization of the master variable fixation strategy is conducive to finding a better upper bound. While it cannot be definitively said that the master variable fixation strategy will not exclude prospective optimal solutions, this strategy is often advantageous for searching for better optimal solutions.

Table A1. The frequency of adopting the master variables fixation strategy.

No.	Instances $\|O\|$-$\|R\|$-$\|L\|$	Frequency		
		First Calculation	Second Calculation	Third Calculation
1	15-4-5	0	0	0
2	20-6-8	0	0	0
3	28-8-10	0	1	1
4	35-10-13	1	0	1
5	76-14-18	3	3	2
6	89-19-25	2	2	3
7	144-25-32	3	3	4
8	137-30-39	4	5	4
9	166-34-44	4	5	3
10	201-40-52	5	4	4

Appendix B

Table A2. Results of small and medium-sized instances.

No.	Instances $\|O\|$-$\|R\|$-$\|L\|$	Gurobi		MABBD			MA	IPGA
		Obj	Gap	UB	LB	Gap	Obj	Obj
1	15-4-5	85.94	0%	85.94	85.94	0.00%	85.94	85.94
2	20-6-8	159.95	0%	159.95	159.95	0.00%	159.95	159.95
3	28-8-10	219.84	0%	219.84	219.84	0.00%	219.84	219.84
4	35-10-13	254.45	10.80%	254.45	254.45	0.00%	254.45	256.5
5	76-14-18	444.62	32.37%	432.98	410.55	5.18%	432.98	438.02
6	89-19-25	803.93	51.79%	755.1	686.42	9.10%	755.1	775.3
7	144-25-32	1622.37	61.34%	1546.4	1341.73	13.24%	1546.4	1561.9
8	137-30-39	1878.61	70.87%	1755.16	1561.29	11.05%	1757.33	1783.01
9	166-34-44	2590.65	73.67%	2309.2	1982.43	14.15%	2293.5	2315.44
10	201-40-52	3086.46	81.21%	2933.5	2318.87	20.95%	2904	2967.13
Average	91-19-25	1114.68	38.21%	1045.25	902.15	7.37%	1040.95	1056.30

Appendix C

In this section, we analyze and process the results of large-sized instances to reflect the advantages of MA more clearly. Owing to the inherent properties of RLOP, the numerical values of the results are relatively large, making it challenging to visually perceive the optimization effect of MA and IPGA. Therefore, this paper attempts to use the upper bound and the lower bound as the comparison benchmark to perform some mathematical processing on the results to reflect the performance of the algorithm more intuitively.

Specifically, we first calculate the difference between the solution obtained by the algorithm to be compared and the benchmark solution. For example, in the 11th instance, MA-LB = 710.5 is the difference between the result of MA, 13,931.9 (from Table 5), and the lower bound, 13,221.4. Then, we measure the performance of these algorithms by the gap between the MA difference and the IPGA difference, where the formula for obtaining the gap is $\Delta^{B}_{MA-IPGA} = \frac{(MA-B)-(IPGA-B)}{IPGA-B}, B \in \{UB, LB\}$. Similarly, taking the 20th instance as an example, $\Delta^{LB}_{MA-IPGA} = -49.25\%$ can be calculated from $\frac{710.5-1400.0}{1400.0} = -49.25\%$.

Table A3. Results of large-sized instances.

No.	Instances $\|O\|$-$\|R\|$-$\|L\|$	Obj		UB as the Benchmark			LB as the Benchmark		
		UB	LB	MA-UB	IPGA-UB	$\Delta^{UB}_{MA-IPGA}$	MA-LB	LPGA-LB	$\Delta^{LB}_{MA-IPGA}$
11	1176-53-60	15,149.3	13,221.4	−1217.4	−527.9	130.61%	710.5	1400.0	−49.25%
12	1687-75-91	23,720.1	20,749.8	−2018.1	−1040.6	93.94%	952.3	1929.8	−50.65%
13	2346-107-138	49,894.5	41,397.4	−4960.3	−3160.0	56.97%	3536.8	5337.1	−33.73%
14	3335-151-181	88,178.9	70,465.1	−10,297.1	−7724.7	33.30%	7416.7	9989.1	−25.75%
15	4689-205-248	173,687.0	142,586.7	−17,158.5	−9999.6	71.59%	13,941.8	21,100.7	−33.93%
16	5613-249-261	259,147.7	228,530.2	−21,998.7	−18,565.3	18.49%	8618.8	12,052.2	−28.49%
17	6467-283-348	315,204.1	263,514.4	−40,151.9	−32,722.2	22.71%	11,537.8	18,967.5	−39.17%
18	6802-321-408	462,138.5	376,430.4	−60,750.3	−47,495.2	27.91%	24,957.9	38,213.0	−34.69%
19	7992-390-445	685,790.7	548,376.6	−98,039.9	−78,313.9	25.19%	39,374.2	59,100.2	−33.38%
20	9537-416-503	832,881.5	694,387.0	−94,662.8	−76,403.4	23.90%	43,831.8	62,091.1	−29.41%
21	10,585-527-585	1,338,174.2	1,156,072.3	−131,060.0	−106,522.2	23.04%	51,041.9	75,579.7	−32.47%
22	14,090-616-708	2,034,856.7	1,806,201.4	−167,180.4	−133,194.3	25.52%	61,474.9	95,461.0	−35.60%
23	15,036-698-768	2,710,522.2	2,247,832.7	−308,396.4	−234,599.4	31.46%	154,293.1	228,090.1	−32.35%
24	17,687-774-820	3,642,381.7	3,147,313.2	−307,198.6	−216,198.5	42.09%	187,869.9	278,870.0	−32.63%
25	18,715-855-1060	4,506,824.1	3,826,579.7	−486,355.4	−390,244.9	24.63%	193,889.0	289,999.5	−33.14%
26	19,461-927-1177	5,420,598.3	4,597,431.1	−580,513.0	−499,182.6	16.29%	242,654.2	323,984.6	−25.10%
27	23,296-1045-1118	7,652,712.8	6,400,667.1	−896,884.4	−785,652.2	14.16%	355,161.3	466,393.5	−23.85%
28	25,284-1167-1179	9,715,936.3	8,712,437.8	−759,262.9	−687,590.5	10.42%	244,235.6	315,908.0	−22.69%
29	27,430-1288-1378	12,807,938.6	11,483,553.2	−870,058.5	−622,377.3	39.80%	454,326.9	702,008.1	−35.28%
30	29,571-1359-1386	14,743,886.8	13,436,757.3	−909,019.7	−611,023.5	48.77%	398,109.8	696,106.0	−42.81%
Average	12,540-575-643	3,373,981.2	2,960,725.2	−288,359.2	−228,126.9	26.40%	124,896.8	185,129.1	−32.54%

References

1. De Koster, R.; Le-Duc, T.; Roodbergen, K.J. Design and control of warehouse order picking. *Eur. J. Oper. Res.* **2007**, *182*, 481–501. [CrossRef]
2. Leon, J.F.; Li, Y.; Peyman, M.; Calvet, L.; Juan, A.A. A Discrete-Event Simheuristic for Solving a Realistic Storage Location Assignment Problem. *Mathematics* **2023**, *11*, 1577. [CrossRef]
3. Da Costa Barros, Í.R.; Nascimento, T.P. Robotic mobile fulfillment systems: A survey on recent developments and research opportunities. *Robotics Auton. Syst.* **2021**, *137*, 103729. [CrossRef]
4. Wurman, P.R.; D'Andrea, R.; Mountz, M. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. *AI Mag.* **2008**, *29*, 9.
5. Yuan, R.; Graves, S.C.; Cezik, T. Velocity-based storage assignment in semi-automated storage systems. *Prod. Oper. Manage.* **2019**, *28*, 354–373. [CrossRef]
6. Lamballais, T.; Roy, D.; De Koster, M.B.M. Estimating performance in a robotic mobile fulfillment system. *Eur. J. Oper. Res.* **2017**, *256*, 976–990. [CrossRef]
7. Logistics, I.Q. Warehouse Automation Market—A $27 Billion Opportunity by 2025. Available online: https://www.thelogisticsiq.com/research/warehouse-automation-market/ (accessed on 6 February 2023).
8. Weidinger, F.; Boysen, N.; Briskorn, D. Storage assignment with rack-moving mobile robots in KIVA warehouses. *Transp. Sci.* **2018**, *52*, 1479–1495. [CrossRef]
9. Zhuang, Y.; Zhou, Y.; Hassini, E.; Yuan, Y.; Hu, X. Rack retrieval and repositioning optimization problem in robotic mobile fulfillment systems. *Transp. Res. Part E Logist. Transp. Rev.* **2022**, *167*, 102920. [CrossRef]
10. Liu, H.; Wang, F.; Zhao, J.; Yang, J.; Tan, C.; Zhou, L. Performance Analysis of Picking Path Strategies in Chevron Layout Warehouse. *Mathematics* **2022**, *10*, 395. [CrossRef]
11. Hausman, W.H.; Schwarz, L.B.; Graves, S.C. Optimal storage assignment in automatic warehousing systems. *Manag. Sci.* **1976**, *22*, 629–638. [CrossRef]
12. Kuo, R.J.; Kuo, P.H.; Chen, Y.R.; Zulvia, F.E. Application of metaheuristics-based clustering algorithm to item assignment in a synchronized zone order picking system. *Appl. Soft Comput.* **2016**, *46*, 143–150. [CrossRef]
13. Glock, C.H.; Grosse, E.H. Storage policies and order picking strategies in U-shaped order-picking systems with a movable base. *Int. J. Prod. Res.* **2012**, *50*, 4344–4357. [CrossRef]
14. Bortolini, M.; Botti, L.; Cascini, A.; Gamberi, M.; Mora, C.; Pilati, F. Unit-load storage assignment strategy for warehouses in seismic areas. *Comput. Ind. Eng.* **2015**, *87*, 481–490. [CrossRef]
15. Caron, F.; Marchet, G.; Perego, A. Routing policies and COI-based storage policies in picker-to-part systems. *Int. J. Prod. Res.* **1998**, *36*, 713–732. [CrossRef]
16. Li, J.; Moghaddam, M.; Nof, S.Y. Dynamic storage assignment with product affinity and ABC classification—A case study. *Int. J. Adv. Manuf. Technol.* **2016**, *84*, 2179–2194. [CrossRef]
17. Yang, C.L.; Nguyen, T.P.Q. Constrained clustering method for class-based storage location assignment in warehouse. *Ind. Manag. Data Syst.* **2016**, *116*, 667–689. [CrossRef]
18. Pan, J.C.H.; Shih, P.H.; Wu, M.H. Storage assignment problem with travel distance and blocking considerations for a picker-to-part order picking system. *Comput. Ind. Eng.* **2012**, *62*, 527–535. [CrossRef]

19. Xiao, J.; Zheng, L. Correlated storage assignment to minimize zone visits for BOM picking. *Int. J. Adv. Manuf. Technol.* **2012**, *61*, 797–807. [CrossRef]
20. Jane, C.C.; Laih, Y.W. A clustering algorithm for item assignment in a synchronized zone order picking system. *Eur. J. Oper. Res.* **2005**, *166*, 489–496. [CrossRef]
21. Liang, K.; Zhou, L.; Yang, J.; Liu, H.; Li, Y.; Jing, F.; Yang, J. Research on a Dynamic Task Update Assignment Strategy Based on a "Parts to Picker" Picking System. *Mathematics* **2023**, *11*, 1684. [CrossRef]
22. Li, X.; Hua, G.; Huang, A.; Sheu, J.B.; Cheng, T.C.E.; Huang, F. Storage assignment policy with awareness of energy consumption in the Kiva mobile fulfilment system. *Transp. Res. Part E Logist. Transp. Rev.* **2020**, *144*, 102158. [CrossRef]
23. Yuan, R.; Li, J.; Wang, W.; Dou, J.; Pan, L. Storage Assignment Optimization in Robotic Mobile Fulfillment Systems. *Complexity* **2021**, *2021*, 4679739. [CrossRef]
24. Merschformann, M.; Lamballais, T.; De Koster, M.B.M.; Suhl, L. Decision rules for robotic mobile fulfillment systems. *Oper. Res. Perspect.* **2019**, *6*, 100128. [CrossRef]
25. Cai, J.; Li, X.; Liang, Y.; Ouyang, S. Collaborative optimization of storage location assignment and path planning in robotic mobile fulfillment systems. *Sustainability* **2021**, *13*, 5644. [CrossRef]
26. Merschformann, M. Active repositioning of storage units in robotic mobile fulfillment systems. In Proceedings of the Operations Research Proceedings 2017: Selected Papers of the Annual International Conference of the German Operations Research Society (GOR), Freie Universiät Berlin, Berlin, Germany, 6–8 September 2017; Springer International Publishing: Berlin/Heidelberg, Germany, 2018; pp. 379–385.
27. Krenzler, R.; Xie, L.; Li, H. Deterministic pod repositioning problem in robotic mobile fulfillment systems. *arXiv* **2018**, arXiv:1810.05514.
28. Xu, W.; Wang, X.; Guo, Q.; Song, X.; Zhao, R.; Zhao, G.; Yang, Y. Decomposition Is All You Need: Single-Objective to Multi-Objective Optimization towards Artificial General Intelligence. *Mathematics* **2023**, *11*, 4390. [CrossRef]
29. Dulebenets, M.A. An Adaptive Polyploid Memetic Algorithm for scheduling trucks at a cross-docking terminal. *Inf. Sci.* **2021**, *565*, 390–421. [CrossRef]
30. Teck, S.; Dewil, R. A bi-level memetic algorithm for the integrated order and vehicle scheduling in a RMFS. *Appl. Soft Comput.* **2022**, *121*, 108770. [CrossRef]
31. Pan, J.C.H.; Shih, P.H.; Wu, M.H.; Lin, J.H. A storage assignment heuristic method based on genetic algorithm for a pick-and-pass warehousing system. *Comput. Ind. Eng.* **2015**, *81*, 1–13. [CrossRef]
32. Yan, B.; Yan, C.; Long, F.; Tan, X.C. Multi-objective optimization of electronic product goods location assignment in stereoscopic warehouse based on adaptive genetic algorithm. *J. Intell. Manuf.* **2018**, *29*, 1273–1285. [CrossRef]
33. Tu, M.; Yang, M.F.; Kao, S.L.; Lin, F.C.; Wu, M.H.; Lin, C.K. Using a heuristic multi-objective genetic algorithm to solve the storage assignment problem for CPS-based pick-and-pass system. *Enterp. Inf. Syst.* **2021**, *15*, 1238–1259. [CrossRef]
34. Liu, L.; Wang, T. An evolvable hardware method based on elite Partheno-Genetic Algorithm. *Appl. Soft Comput.* **2021**, *113*, 107904. [CrossRef]