symmetry

MDPI

*Article*

# Scheduling Optimization of Compound Operations in Autonomous Vehicle Storage and Retrieval System

**Lili Xu, Jiansha Lu and Yan Zhan ***

College of Mechanical Engineering, Zhejiang University of Technology, Hangzhou 310023, China; 2111602062@zjut.edu.cn (L.X.); ljs@zjut.edu.cn (J.L.)
* Correspondence: yzhan@zjut.edu.cn

**Abstract:** The increasing demand for storing various types of goods has led to a raise in the need for storage capacity in warehousing systems. Autonomous vehicle storage and retrieval systems (AVS/RSs) offer high flexibility by allowing different configurations to meet different storage requirements. The system mainly completes operations through elevators and multiple rail-guided vehicles (RGVs). This paper focuses on the scheduling optimization of compound operations in the AVS/RS to improve system performance. Compound operations involve the coordinated execution of both single-command and double-command operations. A mathematical model with compound operations was proposed and effectively decomposed into a horizontal component for RGVs and a vertical counterpart for the elevator, which can represent the operations of one elevator cooperating with multiple RGVs. The goal of this model was to minimize the makespan for compound operations and to determine the optimal operation sequence and path for RGVs. An improved discrete particle swarm optimization (DPSO) algorithm called AGDPSO was proposed to solve the model. The algorithm combines DPSO and a genetic algorithm in an adaptive manner to prevent the algorithm from falling into local optima and relying solely on the initial solution. Through rigorous optimization, optimal parameters for the algorithm were identified. When assessing the performance of our improved algorithm against various counterparts, considering different task durations and racking configurations, our results showed that AGDPSO outperformed the alternatives, proving its effectiveness in enhancing system efficiency for the model. The findings of this study not only contribute to the optimization of AVS/RS but also offer valuable insights for designing more efficient warehouses. By streamlining scheduling, improving operations, and leveraging advanced optimization techniques, we can create a more robust and effective storage and retrieval system.

**Keywords:** autonomous vehicle storage and retrieval system; compound operations; scheduling optimization; improved discrete particle swarm optimization

check for updates

**Citation:** Xu, L.; Lu, J.; Zhan, Y. Scheduling Optimization of Compound Operations in Autonomous Vehicle Storage and Retrieval System. *Symmetry* **2024**, *16*, 168. https://doi.org/10.3390/sym16020168

Academic Editors: Lei Yue and Jinhua Zhang

Received: 17 December 2023
Revised: 24 January 2024
Accepted: 26 January 2024
Published: 31 January 2024

**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

## 1. Introduction

With the development of e-commerce and automation technology, the requirements of the product variety with less volume, quantity with demand uncertainty, and decreased response time in warehouse automation system are increased [1], which can lead to a more flexible system. In 2022, the Chinese warehouse automation industry was worth USD 4.72 million, compared to USD 2.59 m in 2018. The parcel, durable manufacturing, and general merchandise sectors have been largely responsible for driving this growth, registering Compound Annual Growth Rates (CAGRs) of 22.1%, 22.5%, and 18.4%, respectively. Between 2023 and 2027, the outlook for the market is still rosy as companies continue to look for ways to optimize efficiency on the factory floor, with a CAGR of 13.5% predicted by Interact Analysis [2]. This analysis highlights the importance of researching automated warehousing and the increasing demand for warehouses due to rising customer demand. The focus of this paper was to study the flexible autonomous vehicle storage and retrieval system (AVS/RS). Task scheduling within the system plays a crucial role in determining its

performance [3]. Therefore, this study aimed to enhance the operational efficiency of the system through task scheduling analysis. The aim of the scheduling problem in this study was to minimize task completion time, thus improving overall efficiency.

An AVS/RS consists of storage racks, elevators, and rail-guided vehicles (RGVs) [4]. The storage racks are used to temporarily store stock-keeping units (SKUs). The RGVs are fully automated vehicles that provide horizontal movement to store and retrieve SKUs within the racks. Ekren et al. [5] proposed a novel tier-captive AVS/RS that utilized movable lifts to improve the throughput rate, offering a way to enhance system performance by changing the equipment's operating method. Yang et al. [6] studied the movement of RGVs between two aisles and two bays within an aisle. Lerher et al. [7] studied the movement of RGVs between two bays within an aisle. The elevator is an automated vehicle that provides vertical movement by loading RGVs or SKUs between tiers. Scholars have studied three modes of RGV operation on tiers, including tier-captive [8], tier-to-tier [9], and aisle-to-aisle [10], but these modes have limitations in RGV operations. The operation mode of the RGV in these studies is only a simple reciprocating motion, which restricts the storage type and structure of the racks. The AVS/RS was studied to improve the operation modes of RGV, allowing the RGV to store or retrieve SKUs at both ends of a bay of racks and making the system more flexible to accommodate various requirements.

In the study of the problems in AVS/RS, Azadeh et al. [11] categorized the literature in system analysis, design optimization, and operation planning and control for the robotized and automated warehouse systems, and they presented a clear direction in AVS/RS. To achieve higher efficiency and flexibility of the system, Roy et al. [12] considered the varying cross-aisle location and dwell-point policies in AVS/RS to research the design decisions, which pointed out different policies that influence the system's performance. Ekren [1] determined the number of vehicles on the changed demand environment with the multi-objective optimization of minimizing the average cycle time and energy consumption. Yang et al. [6] established a mathematical model aimed to minimize the total time of operation tasks and used a hybrid particle swarm optimize (PSO) algorithm to optimize the solution. Teppia et al. [13] proposed a new automated multi-deep unit-load storage system of a shuttle-based compact system to reduce operational cost and improve volume flexibility. Manzini et al. [14] proposed a solution that solved the system configuration, which works with multiple deep-storage lanes in AVS/RS. Li et al. [3] proved that the performance of the automated storage and retrieval system (AS/RS) relies on task scheduling. Zhen et al. [15] studied the multiple-equipment scheduling problem with automated guided vehicles, elevators, and shuttles in an automated warehouse. They considered the collaborative work between equipment to formulate a model. As a variant of AS/RS, the performance of AVS/RS also relies on task scheduling. Furthermore, their configuration presents useful guidelines for a system layout to adapt to different cargo storage requirements.

Currently, scholars are researching various systems, such as single-deep [4,16], double-deep [17], and multi-deep [18] systems, and variant systems [19]. Each of these systems offers unique advantages in their specific fields, with the ultimate goal of improving efficiency and reducing costs. In pursuit of this objective and system flexibility, an AVS/RS was studied to store multiple types of SKUs by optimizing the operation path of the RGVs in a warehouse without requiring excessive space. This system involves increasing the depth in the storage racks and aisles in the bays to accommodate multiple types of SKUs. The RGVs are capable of performing cross-aisle, cross-bay, and cross-tier operations using elevators, and the number of RGVs and elevators can be adjusted based on the quantity of tasks and set at any end of the tracks. The placement of only one elevator at the front of the transverse track provides an advantage in terms of swift implementation and application. Once the storage and retrieval tasks are completed by the RGVs, there are two types of operations: single command (SC) [20] and double command (DC) [21]. Wang et al. [22] proposed a multi-tier shuttle warehouse system to address the SC retrieval scheduling method and achieved the optimal task sequence with the aim of minimizing operation time. Dong and Jin [23] formulated the travel time model of SC and DC in a tier-to-tier shuttle-based

storage and retrieval system. They proved that when a system aims to be more responsive to customers, it should operate under SC operations. Conversely, if overall capacity is the priority, then DC operations are much more effective. To optimize task efficiency and align with actual production in AVS/RS, RGVs adopt compound operations, combining SC and DC operations to reduce the storage and retrieval time through the careful planning of task sequences and RGV paths. The compound operation and complex movement of RGVs increase the difficulty of task scheduling, which significantly impacts the operating efficiency of AVS/RS. However, studying the compound operation is crucial, as it better reflects actual warehouse operations. Determining the path and tasks of RGVs can effectively enhance system operation efficiency and address the task scheduling problem.

Extensive research has been conducted on the task scheduling problem in AVS/RS. The large-scale problem has been proven to be an NP-hard problem that can be solved by a meta-heuristic algorithm. Wang et al. [24] focused on the task scheduling problem in a tier-captive AVS/RS and scheduled different devices to reduce the expected cycle time of SC. They proposed a time sequence mathematical model and solved it using the elitist non-dominated sorting genetic algorithm (GA). Izakian et al. [25] designed a discrete particle swarm optimization (DPSO) algorithm to solve the scheduling problem in a job shop, demonstrating the effectiveness of the algorithm. Both the improved GA and DPSO methods have been proven to be effective for solving scheduling problems, highlighting the effectiveness of the proposed improved DPSO.

Large-scale combinatorial optimization (CO) is an NP-hard problem and is considered a promising approach for scheduling problems, which can be efficiently addressed by meta-heuristic algorithms. Frequently utilized meta-heuristic algorithms include GA, differential evolution (DE), simulated annealing (SA), PSO, and ant colony optimization (ACO) [26]. For the research of these algorithms [27], scholars combined two or more methods to improve the advantages and avoid the disadvantages of each algorithm, which want to achieve better optimization results. These meta-heuristic algorithms and their variants have been extensively applied in various fields, such as machinery [28], medicine [29], chemistry [30], etc. The problems solved by these algorithms are combinatorial optimization (e.g., traveling salesman problems [31], job scheduling problems [32]), scheduling problems [33,34], and machine learning problems (e.g., neural network training [35], pattern recognition [36]), multi-objective optimization problems [37,38], etc.) These problem categories are interconnected in some cases, and improved algorithms often enhance solution efficiency and stability, albeit potentially increasing algorithm runtime.

The contribution of this paper is the improvement of the physical structure of traditional AVS/RS by creating a multi-deep system to enhance system flexibility. Multiple RGVs and an elevator can be controlled based on the scale of tasks. The motion characteristics of RGVs are investigated to study the scheduling problem of compound operations in terms of the sequence and path of RGVs. To construct a mathematical model, a compound task was divided into the storage stage, idle load stage, and retrieval stage, and the operation time of RGVs and the elevator was discussed based on whether storage and retrieval tasks were on the same tier. The objective function of the model was established to minimize the makespan of RGVs, and it was solved using an improved discrete particle swarm optimization algorithm. The research consists of the following sections: Section 2 provides a description of AVS/RS, introduces the operation method of RGVs, and presents the problem that we studied. In Section 3, a scheduling optimization model for AVS/RS is established. Section 4 describes the steps of the improved DPSO with the scheduling model. Section 5 presents the experimental setup in AVS/RS to verify the effectiveness of the algorithm and analyze the results. Finally, concluding remarks and future perspectives are provided.

## 2. System Description

The coordinate axis of the AVS/RS is shown in Figure 1. Where the X-axis is positioned in the middle of the first bay, the Y-axis is at the center of the columns, and the Z-axis

represents the vertical direction. The coordinate origin (O) is at the intersection of these axes. The RGV, smaller than the racks, can hoist SKUs to any position within the system by moving along the X-direction tracks, the Y-direction racks, and the vertical direction facilitated by the elevator. The input/output (I/O) point serves as the entrance or exit for the RGV. Different SKUs can be stored in a bay, and two types of SKUs can be stored in bays between two tracks without requiring relocation upon task completion. The AVS/RS demonstrates flexibility, allowing for enhanced operating efficiency through the addition of RGVs and elevators, accommodating multiple types of SKUs.

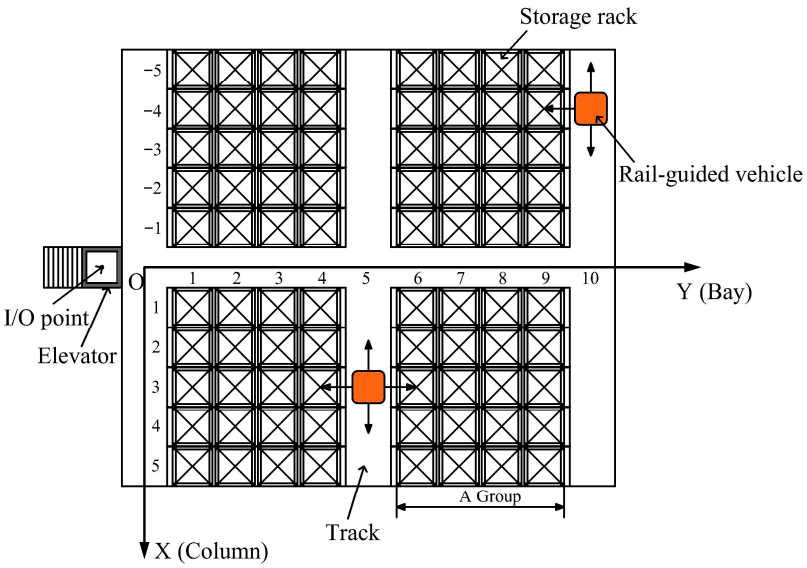


**Figure 1.** Structure diagram of the AVS/RS.

In the system configuration, there are 2$N$ columns, $M$ bays, and $L$ tiers in the storage racks. As shown in Figure 1, there are 10 columns and 10 bays. Four bays form a group, excluding the track. There are $G$ groups and $M = 5\,G$. Each group is separated by tracks, and the bays that are aligned with the X-direction tracks are designated as $(4\,g + 1)$ ($g = 1, 2, \ldots, G$). The inbound/outbound point, serving as the RGV entrance and exit, is the I/O point. Storage racks are symmetrical to the plane formed by the I/O point and the longitudinal tracks. The right racks of the I/O point are labeled as columns 1 to $N$, with the column closest to the I/O point considered the first column. On the left side of the I/O point, racks are labeled as columns $-N$ to $-1$, with the column closest to the I/O point designated as the negative one column. Referring to the coordinate axis in Figure 1, the column closest to the I/O point is the first column, and the bottom layer is the first layer. The coordinates of the $x$ column, $y$ bay, and $z$ tier are represented as $(x, y, z - 1)$ ($x = -N, \ldots, N$, and $x \neq 0$; $y = 1, 2, \ldots, M$, and $y \neq 4\,g + 1$. The tracks exist when the $y$ is equal to $4\,g + 1$; $z = 1, 2, \ldots, L$).

### *2.1. Operation Description*

In the research, both the scheduling problem of the RGVs and the sequence with tasks needs to be studied. RGVs are responsible for completing batches of customer tasks, which comprise SC and/or DC operations. The SC operation involves either storage or retrieval tasks in a command cycle, while the DC operation entails both storage and retrieval tasks in a command cycle. To enhance task efficiency within the AVS/RS, RGVs employ compound operations involving both SC and DC operations. This strategy involves the meticulous planning of task sequences and RGV paths to minimize storage and retrieval times. However, the adoption of compound operations and the intricate movements of RGVs pose challenges to task scheduling, significantly impacting the operational efficiency of the AVS/RS. To illustrate a DC operation, we assume a storage or retrieval location

labeled as A (see Figure 2). The elevator is positioned at the first tier in the system in its initial state. The three stages of a DC operation are as follows:

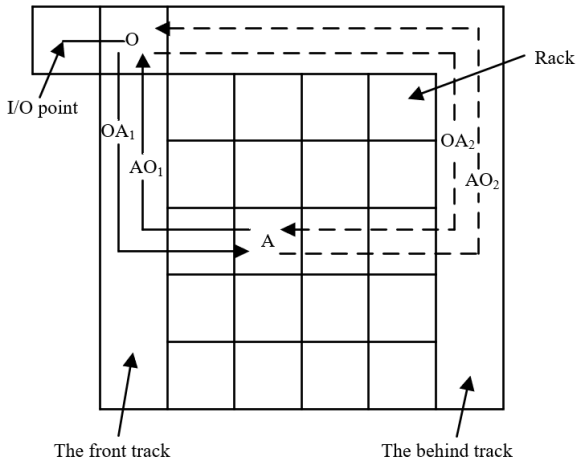


**Figure 2.** Paths of storage and retrieval progress.

1. Storage stage: In this stage, the RGV transports SKUs from the I/O point to storage location A. The RGV picks up SKUs that need to be stored from the I/O point and loads them onto an elevator to the relevant tier. Then, the elevator releases the RGV with SKUs at the O point. The SKUs loaded by the RGV need to be stored in the storage location A. The path OA is selected. If there are no SKUs between A and the front track, then the RGV will follow the path $OA_1$ to A to store the SKUs. If there are SKUs between A and the front track, then the RGV will follow the path $OA_2$ to A to store the SKUs. In addition, the elevator waits on the tier after the RGV leaves.
2. Idle load stage: In this stage, the RGV moves from the storage location to the retrieval location. The RGV is in the idle load stage after completing the storage task. If the storage location and the retrieval location are on the same tier, then the RGV follows the shortest path described in Section 3 to the retrieval location. If the storage location and retrieval location are on different tiers, then the RGV follows the path $AO_1$ to the O point. Then, the elevator loads the RGV and moves to the retrieval tier. After that, the elevator releases the RGV to the O point, and the RGV follows the path $OA_1$ to the retrieval location. The elevator also waits on the tier after the RGV leaves.
3. Retrieval stage: In this stage, the RGV transports SKUs from the retrieval location A to the I/O point. The RGV picks up the SKUs from the retrieval location and moves them to point O along path AO. If there are no SKUs between the retrieval location and the front track, then the RGV follows path $OA_1$ from the retrieval location to point O. If there are SKUs between rack A and the front track, then the RGV follows path $OA_2$ from the retrieval location to point O. Afterwards, the elevator loads the loaded RGV from point O and moves to the I/O point.

### *2.2. Problem Description*

In the AVS/RS, the order sequence of a batch of customer orders should be determined by scheduling. These orders are completed by the elevator and the RGV with compound operations. Specifically, in a batch of customer orders, there are $P$ storage tasks and $Q$ retrieval tasks that are completed by an elevator and $m$ RGVs. To complete these tasks, the RGVs perform operations $K$ times, where $K = \max(P, Q)$. The DC operations are performed $K'$ times, where $K' = \min(P, Q)$. The SC operations, on the other hand, are performed $K''$ times and $K'' = K - K'$. For each task, denoted as $k = 0, 1, \ldots, K$, the storage operation is $p = 0, 1, \ldots, K$, with the relevant rack location being $(x_q, y_q, z_q)$. In the case of a single retrieval operation ($i = 0$), the storage location is set as (0, 0, 1). Similarly, for each retrieval operation denoted as $q = 0, 1, \ldots, K$, the relevant location is $(x_q, y_q, z_q)$. If $q = 0$ denotes

a single storage operation, then the retrieval rack location is (0, 0, 1). Furthermore, it is important to note that $p$ and $q$ cannot both be equal to 0 simultaneously. This is due to the fact that a DC operation consists of three stages: storage stage, idle load stage, and retrieval stage. Throughout these stages, both the elevator and the RGVs carry out $K$ times operations, which results in a total of $3K$ stages. The operations performed by the RGVs are represented by the set $AVS = \{AVS_1, AVS_2, \ldots, AVS_m\}$. Similarly, the operations of the RGV are denoted by the set $S = \{1, 2, \ldots, m\}$, where $AVS_i = \left\{avs_1^i, avs_2^i, \ldots, avs_{E_i}^i\right\}$ corresponds to the operation stage time of $i$th RGV, and $E_i$ is the $i$th RGV's operation stage. $avs_{e_i}^i$ is the $e_i$th time operation stage of the $i$th RGV. The set operation stages for the elevator are given by $ELE = \{ele_1, ele_2, \ldots, ele_{3K}\}$, and $ele_j$ is the $j$th operation stage of the elevator. Additionally, $ST$ signifies the operation stage of the $avs_{e_i}^i$ and the $ele_j$. If the mod of $avs_{e_i}^i/3$ and $ele_j/3$ is 1, then it is a storage stage; if the mod is 2, then it is an idle load stage; if the mod is 0, then it is a retrieval stage. The task allocation and execution sequence of the RGV and the elevator must be arranged reasonably under the given constraints. These constraints are constructed as follows:

$$E_i > 0, i \in [1, 3K], \tag{1}$$

$$\bigcap_{i=1}^{m} avs_{E_i}^i = \varnothing, \tag{2}$$

$$\bigcup_{i=1}^{m} avs_{E_i}^i = 3K, \tag{3}$$

$$ele_s \cap ele_d = \varnothing, \forall s, d \in [1, 3K], s \neq d, \tag{4}$$

$$ele_s \cap ele_q = 3K, \forall s, d \in [1, 3K], s \neq d, \tag{5}$$

where Equation (1) indicates that each RGV must have at least one task; Equations (2) and (3) indicate that a task can only be completed by a single RGV, and all tasks should be completed by RGVs; Equations (4) and (5) indicate that a task can only be completed by the elevator once, and all tasks should be completed by the elevator.

In the real application of a warehouse, the number of storage tasks does not always match the number of retrieval tasks in a batch of customer orders. Therefore, compound operations are necessary to complete these tasks. When assigning tasks, it is crucial to consider the movement of the RGV and aim to achieve a reasonable and efficient cooperation between the elevator and the RGVs. Accordingly, optimizing the operation sequence and path of the RGVs is essential. The key to system optimization lies in establishing a rational compound operation task scheduling model with the total travel time of the RGV as the objective.

To simplify the research process, the main assumptions of the problem are:

1. Requests, storage locations, and retrieval locations are known with certainty. These requests can be completed by the elevator and the RGVs, which means both storage SKUs and retrieval SKUs can be moved to their relevant locations.
2. The width of the track is the same as the rack.
3. The motion modes of the elevator and RGVs are in constant acceleration/deceleration, and the speed is the same when loaded or idle.
4. Pick-up and set-down times are assumed to be constant.
5. In the initial state, both the elevator and the RGVs are at the bottom of the rack.
6. The RGVs are constantly powered.
7. The elevator can carry only one RGV at a time.

## 3. System Modelling

To construct a model in the AVS/RS, it is necessary to analyze the travel processes of both the elevator and the RGVs. As indicated in the aforementioned operational description, the process can be categorized based on whether the storage and retrieval locations are on the same tier or different tiers. The travel processes of the elevator and the RGVs involve both horizontal and vertical movements. In the vertical direction, the elevator executes a cross-tier operation to pick up the RGVs. In the horizontal direction, the elevator releases RGVs from the I/O point to the O point. Some parameters in the system are described as follows: the length, width, and height of a rack are $l$, $b$, and $c$, respectively; the maximum velocity of the elevator is $v_{\max e}$; the acceleration/deceleration of the loaded elevator is $a_e$; the pick-up/set-down time of the elevator is $t_e$; the maximum velocity of the RGV is $v_{\max a}$; the acceleration/deceleration of the RGV is $a_a$; the direction time of the RGV is $t_{dir}$; and the pick-up/set-down time of the RGV is $t_a$. According to the device motivation in the system [39], if the highest speed of the elevator/RGV can be $v_{top}$, which is smaller than $v_{\max}$, then the running time of the elevator/RGV for $d_e/d_a$ is $t = \sqrt{d_e(d_a)/a_e(a_a)}$; otherwise, the time is $t = d_e(d_a)/v_{\max e}(v_{\max a}) + v_{\max e}(v_{\max a})/a_e(a_a)$.

The decision variable is:

$$x_{k,i} = \begin{cases} 1, \text{ the task } k \text{ is completed by RGV } i; \\ 0, \text{ otherwise.} \end{cases} \tag{6}$$

The objective function is:

$$\min(\max(T_1, T_2, \cdots, T_i, \cdots, T_m)), \tag{7}$$

Subject to:

$$T_i = \sum_{k=1}^{n} x_{k,i}(t_{k,i} + tw_{k,i}), \tag{8}$$

$$\sum_{i=1}^{m} x_{k,i} = 1, k = 1, 2, \cdots, 3K, \tag{9}$$

where Equation (6) indicates that task $k$ is operated by RGV $i$; Equation (7) minimizes the makespan of each RGV; Equation (8) is the operation time of RGV $i$. In the equation, $t_{k,i}$ is the travel time of the elevator loaded with the RGA $i$ complete task $k$; $tw_{k,i}$ is the wait time of the elevator loaded with the RGA $i$ complete task $k$; Equation (9) guarantees the uniqueness of any task when operated once.

The operation time $T_i$ required for RGV $i$ to complete task $k$ depends on the task order. A discussion is needed to calculate the time, and both vertical and horizontal models are necessary to solve for the time.

### *3.1. Horizontal Model*

The completion of task $k$ by the RGVs on a horizontal plane can be divided into three stages as follows.

#### 3.1.1. Storage Stage

The travel path is divided into $OA_1$ and $OA_2$ (see Figure 2), and the travel time is:

$$T_{sap} = \begin{cases} t(\mathrm{AO}_1) \\ t(\mathrm{AO}_2) \end{cases}, \tag{10}$$

$$t(\mathrm{OA}_1) = t(|x_j|l) + t(y_j b) + t_{dir}\,, \tag{11}$$

$$t(\mathrm{OA}_2) = t(|x_p|l) + t((y_p + 6 - \mathrm{mod}(y_p, 5))b) + t((6 - \mathrm{mod}(y_p, 5))b) + 2t_{dir}, \tag{12}$$

where $\mathrm{mod}(y_p, 5)$ is the remainder of $y_p$ divided by 5.

3.1.2. Idle Load Stage

In this stage, the travel path is divided into two situations.

(1) Storage and retrieval racks on the same tier

In this situation, $z_p = z_q$, and there are three paths from storage rack A to retrieval rack B for the RGVs (see Figure 3). The value of $\alpha$ is:

$$\alpha = \begin{cases} 0, & y_p/5 \neq y_q/5 \\ 1, & \text{otherwise} \end{cases},$$

which means that the storage and retrieval coordinates are not in the same group of shelves, and the value of $\beta$ is:

$$\beta = \begin{cases} p, y_p > y_q & \text{and } \mathrm{mod}(y_p, 5) > 2 \\ q, & \text{otherwise} \end{cases},$$

which means that the storage coordinate is greater than the retrieval coordinate, and the storage position is located at the right end of the group. In addition, $y_p/5$ is the integer of $y_p$ divided by 5.

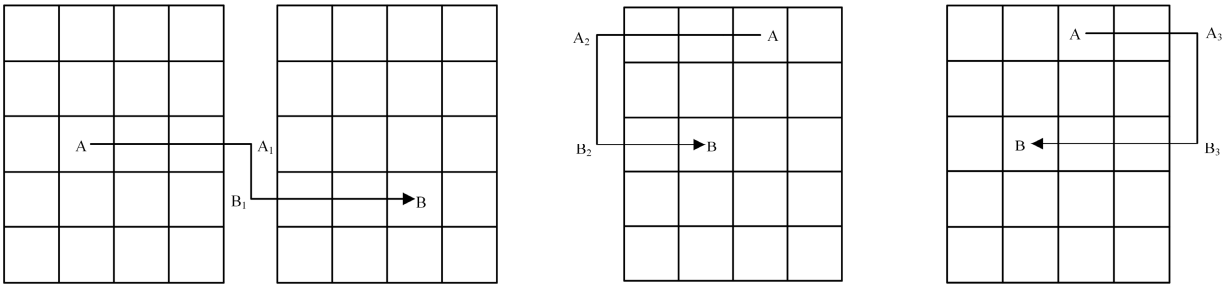


**Figure 3.** The path diagram of the RGV in the idle load stage at $z_i = z_j$. (**a**) $\alpha = 0$; (**b**) $\alpha = 1$ and $\beta = i$; (**c**) $\alpha = 1$ and $\beta = j$.

The travel time of the RGV from storage rack A to retrieval rack B is:

$$T_{iapq} = \begin{cases} t(d(\mathrm{AA_1})) + t(d(\mathrm{A_1B_1})) + t(d(\mathrm{B_1B})) + 2t_{dir} \\ t(d(\mathrm{AA_2})) + t(d(\mathrm{A_2B_2})) + t(d(\mathrm{B_2B})) + 2t_{dir}, \\ t(d(\mathrm{AA_3})) + t(d(\mathrm{A_3B_3})) + t(d(\mathrm{B_3B})) + 2t_{dir} \end{cases} \quad (13)$$

(2) Storage and retrieval racks on different tiers

In this situation, $z_i \neq z_j$, and the travel path is divided into two parts that are the RGV's travel path from the storage location to the buffer and the RGV's travel path from the buffer to the retrieval location. The travel times for these paths are denoted as $T_{iap}$ and $T_{iaq}$, respectively. In addition, as shown in Figure 2, the travel path includes $\mathrm{AO_1}$ and $\mathrm{OA_1}$.

$$T_{iap} = t(|x_p|l) + t(y_p b) + t_{dir}, \quad (14)$$

$$T_{iaq} = t(|y_q b|l) + t(y_q b) + t_{dir}. \quad (15)$$

3.1.3. Retrieval Stage

The travel path is shown in Figure 2, and the path is divided into $\mathrm{AO_1}$ and $\mathrm{AO_2}$. The travel time is found as follows:

$$T_{raq} = \begin{cases} t(\mathrm{AO_1}) \\ t(\mathrm{AO_2}) \end{cases}, \quad (16)$$

$$t(\mathrm{AO}_1) = t(|x_q|l) + t(y_q b) + t_{dir}, \tag{17}$$

$$t(\mathrm{AO}_2) = t(|x_q|l) + t((y_q + 6 - \mathrm{mod}(y_q, 5))b) + t((6 - \mathrm{mod}(y_q, 5))b) + 2t_{dir}. \tag{18}$$

### *3.2. Vertical Model*

When the system completes tasks, the first storage stage for each RGV is completed by the elevator for the first time. This means that stages $ele_1$ to $ele_m$ are storage stages, and these stages are completed by $avs_1^1$ to $avs_m^1$ of each RGV. Additionally, when $p_{ele_1} = 0$, the values of $p_{ele_2}$ to $p_{ele_m}$ correspond to the tiers of $ele_2$ to $ele_m$. $p_{ele_j}$ represents the tier of the elevator starting stage $ele_j$. The stages $ele_{m+1}$ to $ele_{3K}$ are related to the completion time, stage type, and position of the RGVs. The discussion is as follows:

Assuming that the elevator completes stage $ele_j$ and the stage continues to be completed by the $i$th RGV, the stage of RGV is the stage of $avs_{e_i}^i$ when $ele_j = avs_{e_i}^i$. Then,

$$SE_{ele_j} = EE_{ele_{j-1}}, \tag{19}$$

$$SS_{avs_{e_i}^i} = ES_{avs_{e_i-1}^i}, \tag{20}$$

where $SE_{ele_j}$ is the start operation stage time of the elevator in $ele_j$; $EE_{ele_{j-1}}$ is the end operation stage time of the elevator in $ele_{j-1}$; $SS_{avs_{e_i}^i}$ is the start operation stage time of the $i$th RGV in $avs_{e_i}^i$; $ES_{avs_{e_i-1}^i}$ is the end operation stage time of the $i$th RGV in $avs_{e_i-1}^i$.

An RGV and elevator operation model were established for $ele_j$ in the following three situations, and $j$ is bigger than $m$.

#### 3.2.1. Operation Is Storage Stage of the Elevator

In this situation, $ST_{ele_j} = 1$, $ele_{j-1}$ is a retrieval stage and the stage completed by the $i$th RGV, which means $ele_{j-1} = avs_{e_i-1}^i$. After the elevator accomplishes $ele_j$, however, it is not sure that the stage is accomplished by the same RGV, and the operation stage is unknown. The elevator is located in the first tier in the system after completing $ele_{j-1}$. Then,

$$p_{ele_j} = ps_{avs_{e_i}^i} = 1, \tag{21}$$

$$p_{ele_{j+1}} = ps_{avs_{e_i+1}^i} = z_{ele_j}, \tag{22}$$

$$EE_{ele_j} = SE_{ele_j} + t_e + t(z_{ele_j} c), \tag{23}$$

$$ES_{avs_{e_i}^i} = SS_{avs_{e_i}^i} + t(z_{avs_{e_i}^i} c) + T_{ra(ele_j)} + t_a, \tag{24}$$

where $ps_{avs_{e_i}^i}$ is the tier of the $i$th RGV start stage $avs_{e_i}^i$; $z_{ele_j}$ is the tier location of operation stage $ele_j$.

#### 3.2.2. Operation Is Idle Stage of the Elevator

In this situation, $ST_{ele_j} = 2$ and the $i$th RGV accomplish the idle stage $avs_{e_i}^i$. The previous operation stage completed by the elevator was an idle load stage or a storage stage. At this time, the start operation tier of $ele_j$ is the end operation tier of $ele_{j-1}$ for the elevator, which means the following:

$$p_{ele_j} = z_{ele_{j-1}}, \tag{25}$$

$$ps_{avs^i_{e_i}} = z_{avs^i_{e_i-1}}. \tag{26}$$

There are two states of the idle stage:

- The same-tier idle stage ($z_{avs^i_{e_i-1}} = z_{avs^i_{e_i+1}}$): the elevator does not need to complete the idle stage $ele_j$; thus, the start operation tier is the same with the end operation tier, and the tier is located at the start operation tier of $z_{ele_{j+1}}$, which means the following:

$$p_{ele_j} = p_{ele_j+1}, \tag{27}$$

$$EE_{ele_j} = SE_{ele_j}, \tag{28}$$

$$ES_{avs^i_{e_i}} = SS_{avs^i_{e_i}} + T_{ia(avs^i_{e_i-1})(avs^i_{e_i+1})}. \tag{29}$$

- The different-tier idle stage ($z_{avs^i_{e_i-1}} \neq z_{avs^i_{e_i+1}}$): The RGVs must be idle loaded from the storage location to the buffer and then wait in the idle elevator. Subsequently, the RGVs should be transported to the storage tier of the storage location. Once there, the elevator can release the RGVs to the retrieval tier's buffer. Next, the RGVs must move from the buffer to the storage location to complete their idle loading stage.

$$EE_{ele_j} = \max(SE_{ele_j} + t(|p_{ele_j} - z_{avs^i_{e_i-1}}|c), ES_{avs^i_{e_i-1}} + T_{ia(avs^i_{e_i-1})}) + 2t_e + t(|z_{avs^i_{e_i-1}} - z_{avs^i_{e_i+1}}|c), \tag{30}$$

$$ES_{avs^i_{e_i}} = EE_{ele_j} + T_{ia(avs^i_{e_i+1})} + t_a. \tag{31}$$

3.2.3. Operation Is Retrieval Stage of the Elevator

In this situation, $ST_{ele_j} = 0$. Because the next operation stage of the elevator after completing the retrieval stage is the storage stage, $ele_{j-1}$ is not a retrieval stage. Therefore, after the completion of this operation, the elevator layer is $z_{ele_{j-1}}$, which means the following:

$$p_{ele_j} = z_{ele_{j-1}}, \tag{32}$$

$$EE_{ele_j} = \max(SE_{ele_j} + t(|p_{ele_j} - z_{ele_j}|c), SS_{avs^i_{e_i}} + t_e + T_{ra(ele_j)}) + t_a + t((z_{avs^i_{e_i}} - 1)c) + t_e, \tag{33}$$

$$ps_{avs^i_{e_i}} = z_{avs^i_{e_i}}, \tag{34}$$

$$ES_{avs^i_{e_i}} = EE_{ele_j}. \tag{35}$$

Through the above analysis, the total time $T$ for the system to complete $3K$ stage by the elevator and $m$ RGVs is denoted as $EE_{ele_{3K}}$.The time is also the makespan of the RGV, which means the following:

$$\max(T_1, T_2, \cdots, T_m) = EE_{ele_{3K}} \tag{36}$$

In addition, the main objective is to find the minimum makespan of tasks, which is to find the best combination of $i$ and $j$. To minimize the task completion time of the system, the objective function is defined as follows:

$$T = EE_{ele_{3K}} \tag{37}$$

## 4. Solution Algorithm

### *4.1. Discrete Particle Swarm Algorithm*

Based on research on metaheuristic algorithms, PSO is well-established and has demonstrated effectiveness in addressing CO problems. There are various PSO variants, such as parameter tuning, learning exemplar selection, and hybridization strategy [40].

Therefore, we improved upon the PSO algorithm to solve the problem in the formulated model. PSO was introduced by Kennedy and Eberhart in 1995 as a population-based search algorithm inspired by bird flocking and fish schooling behavior [41]. To apply PSO to a discrete problem, Clerc [42] rewrote the PSO equations as DPSO as follows:

$$v_{t+1} = c_1 v_t \oplus c_2(p_{i,t} - x_t) \oplus c_3(p_{g,t} - x_t), \tag{38}$$

$$x_{t+1} = x_t + v_{t+1}, \tag{39}$$

where $v_t$ is velocity at time step $t$; $x_t$ is position at iteration $t$; $p_{i,t}$ is best previous position at iteration $t$; $p_{g,t}$ is the neighbor's best previous position at iteration $t$; $c_1$, $c_2$, $c_3$ are social/cognitive confidence coefficients. In the equations, the author considered $c_2 = c_3$.

An operator $v$ is defined by the following:

$$v = ((i_k, j_k)), k \uparrow_1^{||v||}, \tag{40}$$

which means "exchange numbers $(i_1, j_1)$, then numbers $(i_2, j_2)$, etc., and at last numbers $(i_{||v||}, j_{||v||})$". The opposite of velocity can be found as follows:

$$\neg v = ((i_k, j_k)), k \downarrow_1^{||v||} = ((i_{||v||-k+1}, j_{||v||-k+1})), k \uparrow_1^{||v||}, \tag{41}$$

which means "to do the same transpositions as in $v$, but in reverse order", and $\neg\neg v = v$.

The position $x_{t+1}$ in (33) is found by applying the first transposition of $v_{t+1}$ to $x_t$ and then the second one to the result, etc.

As shown in (33), a position plus a velocity is a position; the value of a position of $p_{i,t}$ minus a position $x_t$ in (32) is a velocity.

$v_1$ and $v_2$ are two velocities. To compute $v_1 \oplus v_2$, the list of transpositions is considered, which contains first those of $v_1$, followed by those of $v_2$.

The value of a real coefficient ($c$, $c \in [0, 1]$) times a velocity $v$ is defined as:

$$cv = ((i_k, j_k)), k \uparrow_1^{c||v||}. \tag{42}$$

When $c > 1$, it holds that $c = k + c'$, where $k \in N^*$, and $c' \in [0, 1]$. The value is defined as follows:

$$cv = v \oplus v \oplus \cdots \oplus v \oplus c'v, \tag{43}$$

and the number of times $vv$ is $k$.

### *4.2. Improved Discrete Particle Swarm Optimization*

The effectiveness of the PSO algorithm heavily depends on its parameters. Employing adaptive parameter methods can enhance efficiency [43]. An improved DPSO algorithm named AGDPSO was proposed to address the CO model in Section 3. AGDPSO combines DPSO and GA in an adaptive manner to prevent the algorithm from converging to local optima and relying solely on the initial solution. Experimental procedures are suggested to identify a suitable parameter combination for the AGDPSO algorithm to fit the CO model. In the algorithm, particles are designed to represent a sequence of orders in the system, and the particle update method must be modified. Algorithm 1 AGDPSO is presented in the algorithm to solve our problem, and its main steps are as follows:

**Algorithm 1** AGDPSO

**Input**: $Pop$, $t = 1$, $P_s$, $P_c$, $P_m$, $c_{\max}$, $c_{\min}$
$Size = |Pop|$; /*$Size$ is the population size of current population $Pop$. */
if $t <= Iter$; /* $Iter$ is the maximum iteration. */
    for $i = 1$ to $Size$
        Calculate the fitness of particles, find the $p_{i,t}$ and $p_{g,t}$.
        Update $c_1$, $c_2$, and $c_3$ by (37).
        If rand() < $c$
            $cv_{i,t} = (1, 1)$;
        else
            $cv_{i,t} = (i_k, j_k)$;
        end
        Update $v_t$ and $x_t$ by (31) and (32).
        Update particles by $P_s$, $P_c$, $P_m$.
    end
else
**return** particles
end
**Output**: fitness

Step 1: Sequence rules

To minimize task completion time, it is recommended to ensure that the storage and retrieval locations are on the same tier when an RGV (rail-guided vehicle) completes a compound operation. This optimization reduces the operation time of the RGV and eliminates idle time for the elevator. There are two rules of the sequence:

- The elevator sequentially completes the first storage stage for all RGVs before proceeding with subsequent tasks to ensure that all RGVs are in the shelves and operational.
- The elevator selects the nearest RGV for operation.

Step 2: Initialize the state of the storage racks

It is recommended to initialize the storage state of the storage racks in the AVS/RS. When an RGV completes a compound operation, it updates the storage status on the relevant rack. The scheduling problem focuses on the RGV path with orders. Rack states should be initialized before scheduling; if the rack is loaded with SKU, then the rack status is 1; otherwise, it is 0.

Step 3: Initialize confidence coefficients

As shown in (31), the confidence coefficients of $c_1$, $c_2$, and $c_3$ need to be determined in the DPSO algorithm. To ensure the diversity of particles, the adaptive weights of $c_1$, $c_2$, and $c_3$ are formulated to find a better solution to our problems than (37):

$$c = \begin{cases} c_{max} - \frac{(c_{max} - c_{min})(f - f_{avg})}{f_{avg} - f_{min}}, & f \le f_{avg} \\ c_{max}, & f > f_{avg} \end{cases}, \tag{44}$$

where $c_{max}$ and $c_{min}$ are the maximum and minimum of the adaptive weight; $f$ is the current fitness value of the particle; $f_{avg}$ is the current average fitness value of all particles; $f_{min}$ is the current minimum fitness value of all particles.

Step 4: Initialize particle population

The positional solutions of particles are encoded in a $3K$ vector and five layers, in which $K$ is the compound operation time of the RGVs. The first layer represents the operation stages. The locations of the vector elements at $(3k - 2)$ denote storage stages, and they are natural numbers included in the range $[0, K]$, where the number 0 is the max $(0, K - P)$. Locations at $(3k - 1)$ denote idle stages, and they are the number of combinations of $(3k - 2)$ and $3k$. The locations at $3k$ denote retrieval stages, and they are natural numbers included in the range $[0, K]$, where the number 0 is the max $(0, K - Q)$. The second layer represents the number of RGVs. As three stages are completed by the same RGV, their values are grouped into three random integers ranging from 1 to $m$. The third layer is

the operation sequence of the elevator, determined according to Step 1 and the number of the first and second layers. The fourth layer is the operation stage of the elevator (1 for storage stage; 2 for idle stage; 3 for retrieval stage). The fifth layer is the number of RGVs when the elevator operation sequence is shown in the third layer. When updating particles, only the first and second layers are transformed, and the last three layers are calculated using the first two layers according to Step 1. The update locations are at $(3k-2)$ or $3k$, corresponding to storage or retrieval stages.

Assuming that $x_{i,t}$ shows the position of the $i$th particle at iteration $t$ and as in Figure 4, the particle undergoes four compound operations, four storage tasks, and three retrieval tasks. Specifically, storage 3 corresponds to retrieval 1, and storage 1 corresponds to retrieval 2 and the storage completion of 2. First, the elevator, loaded with RGV 2, completes storage stage 3. Next, RGV 2 completes idle stage 31, and the elevator returns to the first tier to load RGV 1 for storage stage 1. The elevator waits for RGV 1 to complete idle stage 12 before continuously completing the idle stage. Subsequently, RGV 1 and the elevator jointly complete retrieval stage 2.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1st layer | 3 | 31 | 1 | 1 | 12 | 2 | 2 | 20 | 0 |
| 2nd layer | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 |
| 3rd layer | 3 | 1 | 12 | 2 | 2 | 20 | 0 | 31 | 1 |
| 4th layer | 1 | 1 | 2 | 3 | 1 | 2 | 3 | 2 | 3 |
| 5th layer | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 |

**Figure 4.** A particle position example.

The velocity of particles represents the change sequence of their positions. The elements of the velocity are natural numbers in the range [1, $K$]. Assuming that $v_{i,t}$ shows the velocity of the $i$th particle at iteration $t$, $v_{i,t} = ((3,4),(2,3))$ means the change sequence of the particle's position. The change method involves swapping the position of 3 and 4, then 2 and 3. The variables $p_{i,t}$ and $p_{g,t}$ are defined in (31).

Step 5: Calculate fitness value

The objective of the scheduling problem is to minimize the makespan of the RGV when completing compound operations. The time taken by RGV to complete a batch of an order is denoted as $T$ (as mentioned in (30)), and the fitness function can be expressed as follows:

$$fitness = \min T. \tag{45}$$

Step 6: Update particles

The main update method of particles is shown in (31) and (32). An example is given to clearly describe the update, such as $c_1$ = 0.8, $c_2$ = 0.5, $c_3$ = 0.3, $v_{i,t}$ = ((3,4), (2,3)), $x_{i,t}$ = (1 3 5 4 2 1 3 0 4 2), $p_{i,t}$ = (1 3 4 5 2 1 3 0 4 2), $p_{g,t}$ = (4 3 1 5 2 1 3 0 4 2). If a random number is generated that is less than 0.8, $c_1 v_{i,t}$ = ((3,4), (2,3)); otherwise,$c_1 v_{i,t}$ = (1,1). If a random number is generated that is less than 0.5, $c_2(p_{i,t} - x_{i,t})$ = (3,4); otherwise, $c_2(p_{i,t} - x_{i,t})$ = (1,1). If a random number is generated that is less than 0.3, $c_3(p_{g,t} - x_{i,t})$ = ((1,3), (1,4)); otherwise, $c_3(p_{g,t} - x_{i,t})$ = (1,1). If the section of three random numbers is less than these three confidence coefficients, respectively, $v_{i,t+1}$ = ((2,3), (1,3), (1,4)), $x_{i,t+1}$ = (4 5 1 3 2 1 t3 0 4 2).

Additionally, GA is integrated with the DPSO to diversify particle types. Xu et al. [44] proposed a combination GA and PSO, referred to as GPSO, to mitigate premature convergence and local minima. In the DPSO, named AGDPSO, genetic algorithm elements, such as selection and crossover, are employed to enhance particle diversity. The primary focus is on updating the $x_t$. The $x_{t+1}$ in DPSO is solved by (32), and next, the $x'_{t+1}$ in AGDPSO needs to be solved. The position of particles can express the chromosomes in GA. Elitism

and a random selection method are used for particle selection [45], and a multi-point crossover [46] is employed to modify particles, thereby formulating the AGDPSO.

In the description of the AGDPSO, the population size is assumed to be *Pop*, and the maximum number of iterations is *Iter*. After analysis, the computational complexity of the AGDPSO is summarized as O(population initialization) + O(particle update). The complexity of population initialization is O(*Pop*). During the process of the particle update, although the particle positions of the AGDPSO are encoded in a $3K$ vector and five layers, the primary focus is on the storage stage or retrieval stage of the first layer. In short, O(AGDPSO) = O(*Pop*) + *Iter* × (O(*Pop* × *K*) + O(*Pop*) + O($Pop^2$)). The improvement of DPSO did not alter the complexity.

## 5. Simulation Analysis

The software of MATLAB 2016b was used to solve the scheduling problem in the AVS/RS, and the results were analyzed using an Intel(R) Core$^{TM}$ i7-9750H CPU @ 2.6 GHz, RAM 8.00 GB PC. To verify the validity of the model and find the optimization parameters of the AGDPSO and appropriate configuration in the AVS/RS, some experiments were proposed and analyzed. The population size, select rate, crossover rate, and mutation rate were set as 100, 0.5, 0.8, and 0.2, respectively. The rack in the AVS/RS had five tiers, 20 columns, and 10 bays. The length, width, and height of a rack were 1m, 1 m, and 0.8 m, separately. The maximum velocity of the RGV ($v_{maxa}$) was 2 m/s; the direction change time of RGV ($t_{dir}$) was 3 s; and the acceleration/deceleration ($a_a$) was 1 m/s$^2$. The maximum velocity of the elevator ($v_{maxe}$) was 1 m/s, and the acceleration/deceleration ($a_e$) was 0.5 m/s$^2$. The pick-up/set-down time of the RGV ($t_a$) was 1.5 s, and the pick-up/set-down time of the elevator ($t_e$) was 1.5 s [47].

### *5.1. Parameter Determination and Analysis*

The Automated Storage and Retrieval System (AVS/RS) performed storage tasks 25 times and retrieval tasks 20 times for the experiments. The relevant positions are shown in Table 1. The rack status in the AVS/RS was initialized, and the axis positions of loading SKUs are assumed in Table 2. Other positions consist of idle racks (storage racks should be free, and retrieval racks should contain loaded SKUs). Furthermore, the assigned tasks could be performed, and an RGV was used to simplify the analysis.

Due to the randomness of AGDPSO, each of the tests were run 30 times. Nine sets of [$c_{1\min}$, $c_{1\max}$] were selected to find the optimization range of $c_1$, and seven sets of [$c_{2\min}$, $c_{2\max}$] and [$c_{3\min}$, $c_{3\max}$] were selected to find the optimization range of $c_2$ and $c_3$, respectively. Since $c_1$ = [0.2, 0.8] has been studied [48], $c_1$ = [0.2, 1.0] was tested to find a more suitable range. The values of $c_2$ = [0.2, 0.8] and $c_3$ = [0.2, 0.8] were selected because their values were studied to be 0.5 [25]. In Table 3, Avg is the average optimal object value of 20 times, and Std is the standard deviation of the optimal objective value. The values of $c_2$ and $c_3$ were both 0.5, and the minimized Ave was obtained by $c_1$ = [0.4, 0.6]. According to tests used to further determine the range of the other two parameters of $c_2$ and $c_3$, the minimized Ave was obtained by $c_2$ = [0.4, 0.8] and $c_3$ = [0.4, 0.6], which are shown in Tables 4 and 5, respectively. Additionally, the value of Std indicated the sensitivity of these parameters. Although the obtained range of minimized Ave in Tables 3 and 4 was not the smallest, it was close to the minimized Std. In Table 5, the range of minimum Ave was consistent with the range of the minimum Std. Upon analyzing the parameters, it became evident that a wide parameter value range, whether large or small, led to an increase in the STD and the Ave. We chose the range of minimized Ave and Std to solve the problem. Therefore, $c_1$ = [0.4, 0.6], $c_2$ = [0.4, 0.8], and $c_3$ = [0.4, 0.6] were selected to solve the researched problem.

**Table 1.** Storage and retrieval rack list.

| Number | Storage Rack | Number | Retireval Rack |
|---|---|---|---|
| 1 | (3, 4, 5) | 1 | (5, 2, 5) |
| 2 | (−3, 4, 5) | 2 | (5, 4, 2) |
| 3 | (6, 9, 3) | 3 | (5, 3, 3) |
| 4 | (5, 7, 2) | 4 | (3, 17, 1) |
| 5 | (6, 18, 4) | 5 | (4, 7, 5) |
| 6 | (5, 7, 4) | 6 | (−4, 4, 4) |
| 7 | (2, 14, 3) | 7 | (5, 3, 4) |
| 8 | (−10, 2, 2) | 8 | (5, 2, 1) |
| 9 | (1, 9, 3) | 9 | (1, 8, 3) |
| 10 | (8, 7, 4) | 10 | (7, 19, 2) |
| 11 | (8, 8, 5) | 11 | (8, 9, 3) |
| 12 | (5, 2, 3) | 12 | (8, 3, 3) |
| 13 | (2, 7, 1) | 13 | (2, 17, 2) |
| 14 | (2, 7, 4) | 14 | (9, 7, 4) |
| 15 | (−7, 7, 3) | 15 | (6, 10, 4) |
| 16 | (−7, 15, 1) | 16 | (−10, 19, 2) |
| 17 | (−1, 7, 3) | 17 | (2, 7, 2) |
| 18 | (−7, 7, 5) | 18 | (−5, 7, 5) |
| 19 | (10, 7, 4) | 19 | (9, 19, 5) |
| 20 | (−4, 12, 3) | 20 | (−4, 17, 3) |
| 21 | (4, 8, 5) | | |
| 22 | (3, 8, 5) | | |
| 23 | (−9, 17, 4) | | |
| 24 | (7, 12, 4) | | |
| 25 | (−7, 3, 5) | | |

**Table 2.** Racks with loading SKUs.

| Number | Position | Number | Position | Number | Position | Number | Position |
|---|---|---|---|---|---|---|---|
| 1 | (3, 3, 5) | 7 | (2, 13, 3) | 13 | (−10, 2, 3) | 19 | (5, 3, 3) |
| 2 | (5, 7, 3) | 8 | (7, 7, 4) | 14 | (6, 17, 4) | 20 | (−10, 3, 4) |
| 3 | (3, 15, 1) | 9 | (2, 5, 3) | 15 | (5, 2, 2) | 21 | (5, 2, 1) |
| 4 | (4, 2, 4) | 10 | (5, 5, 3) | 16 | (5, 3, 4) | 22 | (4, 7, 5) |
| 5 | (1, 10, 3) | 11 | (5, 2, 4) | 17 | (3, 17, 1) | 23 | (5, 5, 2) |
| 6 | (2, 14, 3) | 12 | (9, 9, 3) | 18 | (5, 2, 5) | 24 | (−4, 4, 4) |

**Table 3.** Results of the AGDPSO algorithm under different ranges of $c_1$.

| $c_{1min}$ | $c_{1max}$ | Avg. (s) | Std. |
|---|---|---|---|
| 0.2 | 0.4 | 972.35 | 6.02 |
| | 0.6 | 965.23 | 7.14 |
| | 0.8 | 1027.37 | 6.35 |
| | 1.0 | 981.50 | 8.71 |
| 0.4 | 0.6 | 947.14 | 2.83 |
| | 0.8 | 956.32 | 3.28 |
| | 1.0 | 963.10 | 5.74 |
| 0.6 | 0.8 | 991.41 | 2.32 |
| | 1.0 | 1005.00 | 4.03 |
| 0.8 | 1.0 | 983.55 | 4.20 |

**Table 4.** Results of the AGDPSO algorithm under different ranges of $c_2$.

| $c_{2min}$ | $c_{2max}$ | Avg. (s) | Std. |
|---|---|---|---|
| 0.2 | 0.4 | 976.60 | 4.51 |
| | 0.6 | 968.21 | 5.35 |
| | 0.8 | 982.16 | 6.74 |
| 0.4 | 0.6 | 1021.70 | 4.60 |
| | 0.8 | 940.82 | 3.98 |
| 0.6 | 0.8 | 963.24 | 6.12 |
| 0.5 | | 956.32 | 3.03 |

**Table 5.** Results of the AGDPSO algorithm under different ranges of $c_3$.

| $c_{3min}$ | $c_{3max}$ | Avg. (s) | Std. |
|---|---|---|---|
| 0.2 | 0.4 | 997.11 | 4.42 |
| | 0.6 | 963.59 | 6.25 |
| | 0.8 | 993.80 | 6.61 |
| 0.4 | 0.6 | 931.32 | 3.23 |
| | 0.8 | 945.73 | 5.73 |
| 0.6 | 0.8 | 951.90 | 5.07 |
| 0.5 | | 940.82 | 3.68 |

### *5.2. Performance Analysis*

Previous research indicated that AGDPSO can effectively solve the scheduling problem of AVS/RS. Three algorithms of the modified discrete shuffled frog leaping algorithm (MDSFLA) [49], discrete version of the improved grey wolf optimizer (DI-GWO) [50], and genetic Tabu search algorithm with neighborhood clipping (GTS_NC) [51] were employed to solve the same problems, and their performances were compared with the AGDPSO. Three scenarios involving different numbers of RGVs, various operation times, and racking configurations in AVS/RS were considered to evaluate the algorithm's performance.

#### 5.2.1. Number of RGVs

In this section, 1, 2, 3, and 4 RGVs were used to complete the tasks in Table 1. Results from the four algorithms are shown in Table 6, and the iteration diagram is shown in Figure 5. It can be seen that the values of Avg were the best by the AGDPSO, except when the number of RGV was 1, and the AGDPSO demonstrated the most stability. Furthermore, with a single RGV, as the number of RGVs increased, the efficiency of the AGDPSO improved to 20.95%, 26.31%, and 30.13%. Although efficiency improved, the rate of improvement gradually slowed down, suggesting that the number of RGVs was not necessarily the key factor for optimal performance in AVS/RS. Finding a suitable number of RGVs based on actual requirements is crucial.

#### 5.2.2. Operation Times

In this section, DC operation times of three batch orders were 25, 50, and 100 in the AVS/RS. Randomly generated relevant positions of these orders were solved using four algorithms to determine RGV operation times. The results, as shown in Table 7, indicate that the AGDPSO yielded the best optimization result. Specifically, at operation times of 25, AGDPSO reduced operation time by 2.88%, −0.24%, and 2.08% compared to MDSFL, DI-GWOCD, and GTS_NC, respectively. At the operation times of 50, compared to the other algorithms, AGDPSO reduced the operation time by 9.13%, 5.33%, and 10.04%, respectively. At the operation time of 100, compared to the other algorithms, AGDPSO reduced the operation time by 7.54%, 6.19%, and 10.55%, respectively. Similar trends were observed

at operation times of 50 and 100. The iterative process of different algorithms solving operation times of 25, 50, and 100 is illustrated in Figure 6.

**Table 6.** Results of different numbers of RGVs under different algorithms.

| Number of RGVs | Algorithm | Avg. (s) | Std. |
|---|---|---|---|
| 1 | AGDPSO | 931.32 | 2.23 |
| | MDSFLA | 958.96 | 3.44 |
| | DI-GWO | 929.06 | 5.71 |
| | GTS_NC | 951.15 | 3.36 |
| 2 | AGDPSO | 736.22 | 2.55 |
| | MDSFLA | 753.91 | 5.51 |
| | DI-GWO | 752.31 | 7.80 |
| | GTS_NC | 772.83 | 7.65 |
| 3 | AGDPSO | 686.3 | 1.44 |
| | MDSFLA | 691.35 | 7.10 |
| | DI-GWO | 719.7 | 6.05 |
| | GTS_NC | 715.17 | 4.14 |
| 4 | AGDPSO | 650.69 | 1.21 |
| | MDSFLA | 678.56 | 3.65 |
| | DI-GWO | 688.55 | 3.28 |
| | GTS_NC | 664.3 | 2.87 |

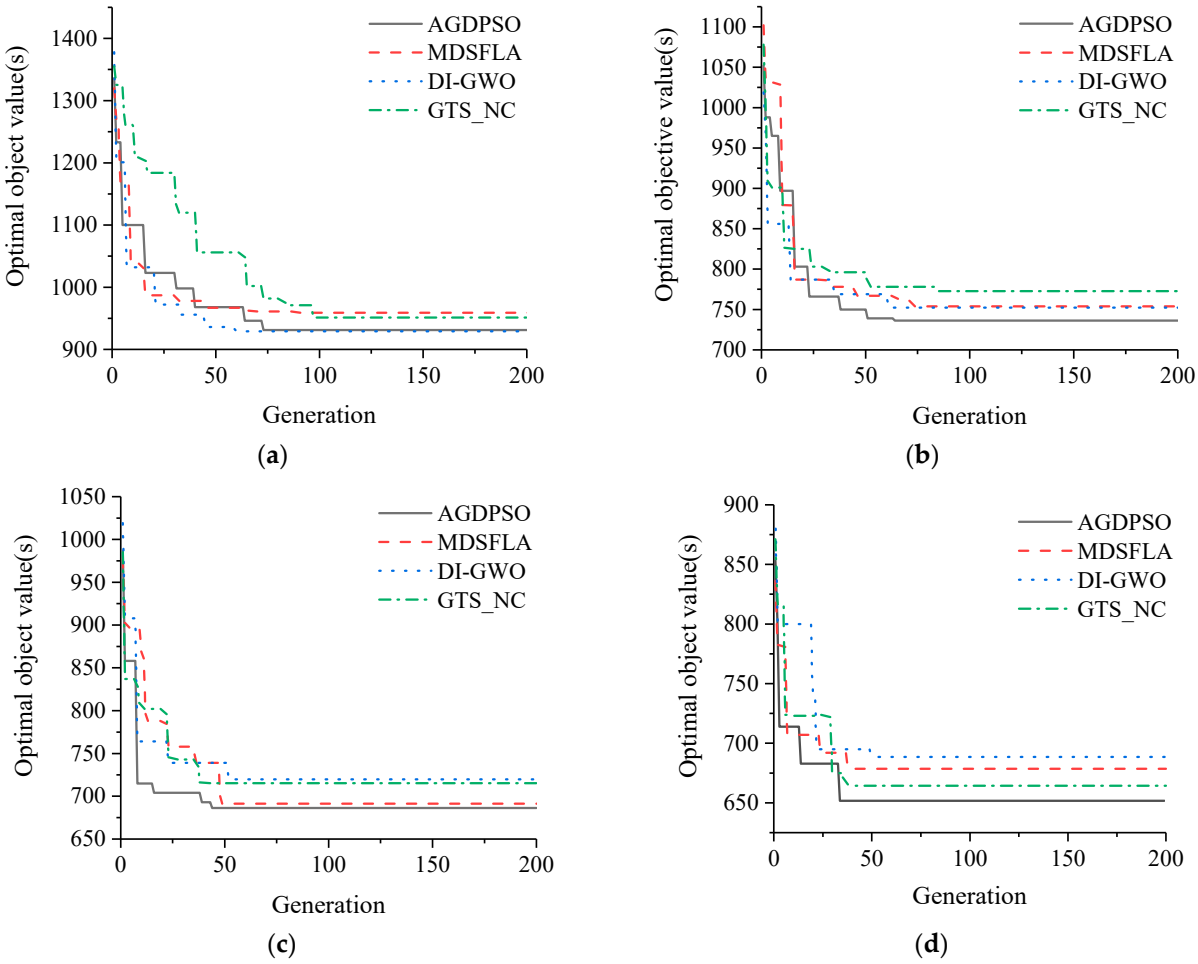


**Figure 5.** Iterative process of different algorithms solves the different numbers of RGV. (**a**) The number of RGVs is 1; (**b**) the number of RGVs is 2; (**c**) the number of RGVs is 4; (**d**) the number of RGVs is 4.

**Table 7.** Results of different algorithms under different operation times.

| Operation Time | Algorithm | Avg. (s) | Std. |
|---|---|---|---|
| 25 | AGDPSO | 931.32 | 2.23 |
| | MDSFLA | 958.96 | 3.44 |
| | DI-GWO | 929.06 | 5.71 |
| | GTS_NC | 951.15 | 3.36 |
| 50 | AGDPSO | 1966.59 | 13.60 |
| | MDSFLA | 2164.21 | 15.40 |
| | DI-GWO | 2077.30 | 26.98 |
| | GTS_NC | 2186.11 | 41.30 |
| 100 | AGDPSO | 4115.70 | 25.65 |
| | MDSFLA | 4451.43 | 72.61 |
| | DI-GWO | 4387.06 | 52.73 |
| | GTS_NC | 4601.25 | 94.46 |

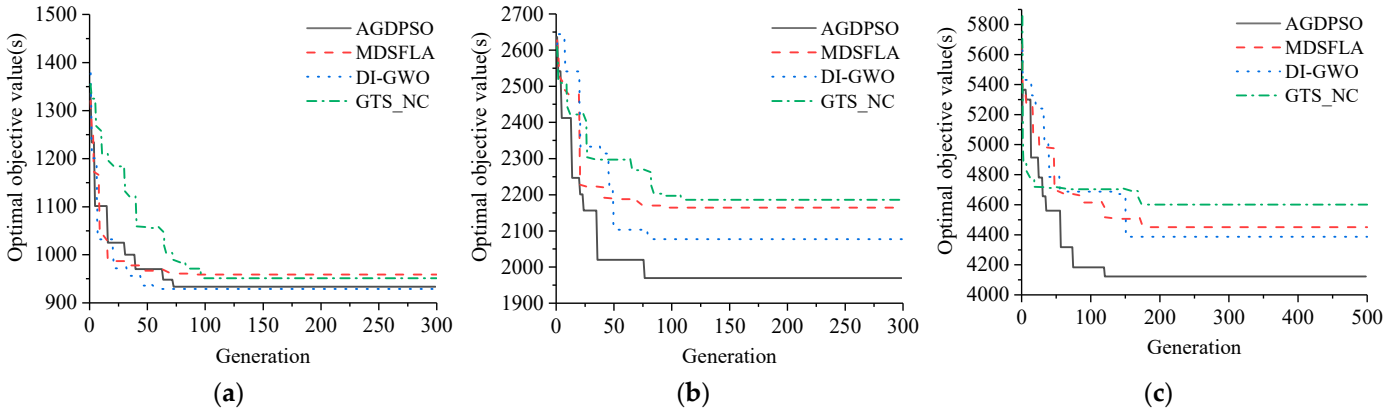


**Figure 6.** Iteration diagram of four algorithms under different operation times, which are 25, 50, and 100. (**a**) The operation time is 25; (**b**) the operation time is 50; (**c**) the operation time is 100.

### 5.2.3. Configurations

In this section, the scheduling problem of different rack configurations in the AVS/RS is studied to provide valuable insights for warehouse designers. Various configurations, such as columns (10 and 20), bays (10 and 20), and layers (4, 5, and 6), were selected. Four algorithms were employed to solve these scenarios, and the results are presented in Table 8. Three deviations of $\Delta_1 = (T_{\text{MDSFLA}} - T_{\text{AGDPSO}})/T_{\text{MDSFLA}}$, $\Delta_2 = (T_{\text{DI-GWO}} - T_{\text{AGDPSO}})/T_{\text{DI-GWO}}$, $\Delta_3 = (T_{\text{GTS\_NC}} - T_{\text{AGDPSO}})/T_{\text{GTS\_NC}}$, were used to show the optimization efficiency of AGDPSO compared to other algorithms. AGDPSO demonstrates clear advantages in most configurations, particularly in situations where the tier is 4 and 5. Although there are instances where AGDPSO performs slightly worse, the impact on results is not significant, reinforcing the superiority of AGDPSO in this context.

As analyzed in the above three sections, AGDPSO can effectively improve operation efficiency in AVS/RS. In the study of the number of RGVs, more RGVs in the system lead to higher operational efficiency. However, with only one elevator in the system, finding a suitable number of RGVs that meet actual requirements is essential, considering constraints such as the cost of RGVs and system operation costs. In the study of racking configurations, higher tiers and larger numbers of columns, bays, and tiers allow for more SKUs to be stored, but they also require more time to complete tasks. Therefore, the system layout should be determined based on storage types to avoid changes that may be difficult to implement in the future.

**Table 8.** Results of different configurations under different algorithms.

| Sequence | $2N \times M \times L$ | $\Delta_1$ (%) | $\Delta_2$ (%) | $\Delta_3$ (%) |
|---|---|---|---|---|
| 1 | $10 \times 10 \times 4$ | 1.9 | 2.6 | 2.3 |
| 2 | $10 \times 10 \times 5$ | 0.4 | 1.2 | 4.3 |
| 3 | $10 \times 10 \times 6$ | 2.0 | −0.6 | 2.2 |
| 4 | $10 \times 20 \times 4$ | 3.6 | 0.5 | 4.4 |
| 5 | $10 \times 20 \times 5$ | 1.7 | 0.4 | 3.3 |
| 6 | $10 \times 20 \times 6$ | 4.2 | 3.1 | 2.4 |
| 7 | $20 \times 10 \times 4$ | 7.4 | 5.3 | 5.4 |
| 8 | $20 \times 10 \times 5$ | 5.5 | 1.7 | 0.7 |
| 9 | $20 \times 10 \times 6$ | 3.8 | −1.3 | 1.1 |
| 10 | $20 \times 20 \times 4$ | 3.0 | 1.2 | 2.4 |
| 11 | $20 \times 20 \times 5$ | 2.2 | 6.5 | 5.3 |
| 12 | $20 \times 20 \times 6$ | −0.3 | 0.9 | 1.8 |

## 6. Conclusions

In this paper, a travel time model of the RGVs and the elevator in the AVS/RS was studied. The task scheduling problem was proposed to enhance the efficiency of the system. The operational methods of the RGVs in the system were introduced to formulate a mathematical model, divided into horizontal and vertical models operated by RGVs and the elevator, respectively. An AGDPSO algorithm was derived to determine the operation sequence and path of the RGVs, with the goal to minimize the makespan. In the solution process, the updated particles were optimized. Experiments were conducted to verify the model and identify suitable parameters, providing better solutions to different scale problems. Finally, various algorithms were used to solve different numbers of RGVs, operation times, and racking configurations in the AVS/RS. Compared to other algorithms, the AGDPSO algorithm required less time to complete operations and increased throughput in the scheduling problem.

This paper did not consider the situation where the RGV faces power depletion, which is a practical concern. The RGV needs to reach the charging station for recharging when its power level is low. In the model, it was assumed that the speed of the elevator and the RGV under load and idle load was the same, which would have an impact on the operating time of the system. Additionally, the predetermined order information during system performance verification and algorithm comparison has inherent result limitations. Thus, future research should incorporate the expected travel time of system operations to assess performance and fulfill storage and retrieval requirements. Moreover, expanding the size of test cases can evaluate the performance of enhanced algorithms across various scenarios.

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

**Funding:** This research was funded by the Zhejiang Science and Technology Plan Project (Grant No. 2018C01003).

**Data Availability Statement:** No new data were created.

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

## References

1. Ekren, B.Y. A multi-objective optimisation study for the design of an AVS/RS warehouse. *Int. J. Prod. Res.* **2021**, *59*, 1107–1126. [CrossRef]
2. Starks, C. Chinese Warehouse Automation Market Set to Grow at CAGR of 13.5% over Next 5 Years. 2023. Available online: https://interactanalysis.com (accessed on 1 December 2023).

3. Li, J.B.; Huang, R.H.; Dai, J.B. Joint optimisation of order batching and picker routing in the online retailer's warehouse in China. *Int. J. Prod. Res.* **2017**, *55*, 447–461. [CrossRef]
4. Malmborg, C.J. Conceptualizing tools for autonomous vehicle storage and retrieval systems. *Int. J. Prod. Res.* **2002**, *40*, 1807–1822. [CrossRef]
5. Ekren, B.Y.; Lerher, T.; Kucukyasar, M.; Jerman, B. Cost and performance comparison of tier-captive SBS/RS with a novel AVS/RS/ML. *Int. J. Prod. Res.* **2023**, 1–15. [CrossRef]
6. Yang, D.; Wu, Y.H.; Huo, D.Y. Research on design of cross-aisles shuttle-based storage/retrieval system based on improved particle swarm optimization. *IEEE Access* **2021**, *9*, 67786–67796. [CrossRef]
7. Lerher, T.; Ekren, Y.B.; Sari, Z.; Rosi, B. Simulation analysis of shuttle based storage and retrieval systems. *Int. J. Simul. Model.* **2015**, *14*, 48–59. [CrossRef]
8. Lerher, T.; Ekren, B.Y.; Dukic, G.; Rosi, B. Travel time model for shuttle-based storage and retrieval systems. *Int. J. Adv. Manuf. Technol.* **2015**, *78*, 1705–1725. [CrossRef]
9. Ha, Y.; Chae, J. A decision model to determine the number of shuttles in a tier-to-tier SBS/RS. *Int. J. Prod. Res.* **2019**, *57*, 963–984. [CrossRef]
10. Lerher, T. Aisle changing shuttle carriers in autonomous vehicle storage and retrieval systems. *Int. J. Prod. Res.* **2018**, *56*, 3859–3879. [CrossRef]
11. Azadeh, K.; De Koster, R.; Roy, D. Robotized and automated warehouse systems: Review and recent developments. *Transp. Sci.* **2019**, *53*, 917–945. [CrossRef]
12. Roy, D.; Krishnamurthy, A.; Heragu, S.; Malmborg, C. Queuing models to analyze dwell-point and cross-aisle location in autonomous vehicle-based warehouse systems. *Eur. J. Oper. Res.* **2015**, *242*, 72–87. [CrossRef]
13. Tappia, E.; Roy, D.; de Koster, R.; Melacini, M. Modeling, analysis, and design insights for shuttle-based compact storage systems. *Transp. Sci.* **2017**, *51*, 269–295. [CrossRef]
14. Manzini, R.; Accorsi, R.; Baruffaldi, G.; Cennerazzo, T.; Gamberi, M. Travel time models for deep-lane unit-load autonomous vehicle storage and retrieval system (AVS/RS). *Int. J. Prod. Res.* **2016**, *54*, 4286–4304. [CrossRef]
15. Zhen, L.; Wu, J.W.; Li, H.L.; Tan, Z.Y.; Yuan, Y.Y. Scheduling multiple types of equipment in an automated warehouse. *Ann. Oper. Res.* **2023**, *322*, 1119–1141. [CrossRef]
16. Kuo, P.H.; Krishnamurthy, A.; Malmborg, C.J. Design models for unit load storage and retrieval systems using autonomous vehicle technology and resource conserving storage and dwell point policies. *Appl. Math. Model.* **2007**, *31*, 2332–2346. [CrossRef]
17. Lerher, T. Travel time model for double-deep shuttle-based storage and retrieval systems. *Int. J. Prod. Res.* **2016**, *54*, 2519–2540. [CrossRef]
18. D'Antonio, G.; De Maddis, M.; Bedolla, J.S.; Chiabert, P.; Lombardi, F. Analytical models for the evaluation of deep-lane autonomous vehicle storage and retrieval system performance. *Int. J. Adv. Manuf. Technol.* **2018**, *94*, 1811–1824. [CrossRef]
19. Jerman, B.; Ekren, B.Y.; Kucukyasar, M.; Lerher, T. Simulation-based performance analysis for a novel AVS/RS technology with movable lifts. *Appl. Sci.* **2021**, *11*, 2283. [CrossRef]
20. Dong, W.Q.; Jin, M.Z.; Wang, Y.Y.; Kelle, P. Retrieval scheduling in crane-based 3D automated retrieval and storage systems with shuttles. *Ann. Oper. Res.* **2021**, *302*, 111–135. [CrossRef]
21. D'Antonio, G.; Chiabert, P. Analytical models for cycle time and throughput evaluation of multi-shuttle deep-lane AVS/RS. *Int. J. Adv. Manuf. Technol.* **2019**, *104*, 1919–1936. [CrossRef]
22. Wang, Y.Y.; Liu, Z.W.; Huang, K.; Mou, S.D.; Zhang, R.X. Model and solution approaches for retrieval operations in a multi-tier shuttle warehouse system. *Comput. Ind. Eng.* **2020**, *141*, 106283. [CrossRef]
23. Dong, W.Q.; Jin, M.Z. Travel time models for tier-to-tier SBS/RS with different storage assignment policies and shuttle dispatching rules. *Transp. Res. Part E Logist. Transp. Rev.* **2021**, *155*, 102485. [CrossRef]
24. Wang, Y.Y.; Mou, S.D.; Wu, Y.H. Task scheduling for multi-tier shuttle warehousing systems. *Int. J. Prod. Res.* **2015**, *53*, 5884–5895. [CrossRef]
25. Izakian, H.; Ladani, B.T.; Abraham, A.; Snasel, V. A discrete particle swarm optimization approach for grid job scheduling. *Int. J. Innov. Comput. Inf. Control.* **2010**, *6*, 4219–4233.
26. Ab Wahab, M.N.; Nefti-Meziani, S.; Atyabi, A. A comparative review on mobile robot path planning: Classical or meta-heuristic methods? *Annu. Rev. Control* **2020**, *50*, 233–252. [CrossRef]
27. Ab Wahab, M.N.; Nefti-Meziani, S.; Atyabi, A. A comprehensive review of swarm optimization algorithms. *PLoS ONE* **2015**, *10*, e0122827. [CrossRef]
28. Lee, C.H.; Shih, K.S.; Hsu, C.C.; Cho, T. Simulation-based particle swarm optimization and mechanical validation of screw position and number for the fixation stability of a femoral locking compression plate. *Med. Eng. Phys.* **2014**, *36*, 57–64. [CrossRef] [PubMed]
29. Mandal, D.; Chatterjee, A.; Maitra, M. Robust medical image segmentation using particle swarm optimization aided level set based global fitting energy active contour approach. *Eng. Appl. Artif. Intell.* **2014**, *35*, 199–214. [CrossRef]
30. Khansary, M.A.; Sani, A.H. Using genetic algorithm (GA) and particle swarm optimization (PSO) methods for determination of interaction parameters in multicomponent systems of liquid-liquid equilibria. *Fluid Phase Equilibria* **2014**, *365*, 141–145. [CrossRef]
31. Attie, O.; Sulkow, B.; Di, C.; Qiu, W.G. Genetic codes optimized as a traveling salesman problem. *PLoS ONE* **2019**, *14*, e0224552. [CrossRef]

32. Zhang, J.; Ding, G.F.; Zou, Y.S.; Qin, S.F.; Fu, J.L. Review of job shop scheduling research and its new perspectives under Industry 4.0. *J. Intell. Manuf.* **2019**, *30*, 1809–1830. [CrossRef]
33. Wang, G.G.; Gao, D.; Pedrycz, W. Solving multiobjective fuzzy job-shop scheduling problem by a hybrid adaptive differential evolution algorithm. *IEEE Trans. Ind. Inform.* **2022**, *18*, 8519–8528. [CrossRef]
34. Houssein, E.H.; Gad, A.G.; Wazery, Y.M.; Suganthan, P.N. Task scheduling in cloud computing based on meta-heuristics: Review, taxonomy, open challenges, and future trends. *Swarm Evol. Comput.* **2021**, *62*, 100841. [CrossRef]
35. Ojha, V.K.; Abraham, A.; Snasel, V. Metaheuristic design of feedforward neural networks: A review of two decades of research. *Eng. Appl. Artif. Intell.* **2017**, *60*, 97–116. [CrossRef]
36. Chuang, L.Y.; Yang, C.S.; Li, J.C.; Yang, C.H. Chaotic genetic algorithm for gene selection and classification problems. *Omics* **2009**, *13*, 407–420. [CrossRef]
37. Verma, S.; Pant, M.; Snasel, V. A comprehensive review on NSGA-II for multi-objective combinatorial optimization problems. *IEEE Access* **2021**, *9*, 57757–57791. [CrossRef]
38. Biswas, S.; Acharyya, S. Multi-objective simulated annealing variants to infer gene regulatory network: A comparative study. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2021**, *18*, 2612–2623. [CrossRef] [PubMed]
39. Ekren, B.Y. A simulation-based experimental design for SBS/RS warehouse design by considering energy related performance metrics. *Simul. Model. Pract. Theory* **2020**, *98*, 101991. [CrossRef]
40. Wei, B.; Xia, X.W.; Yu, F.; Zhang, Y.L.; Xu, X.; Wu, H.R.; Gui, L.; He, G.L. Multiple adaptive strategies based particle swarm optimization algorithm. *Swarm Evol. Comput.* **2020**, *57*, 100731. [CrossRef]
41. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the International Conference on Neural Networks (ICNN'95), Perth, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948.
42. Clerc, M. Discrete particle swarm optimization, illustrated by the traveling salesman problem. In *New Optimization Techniques in Engineering*; Studies in Fuzziness and Soft Computing; Springer: Berlin/Heidelberg, Germany, 2004; Volume 141, pp. 219–239.
43. Liu, H.; Zhang, J.; Zhou, M. Adaptive particle swarm optimizer combining hierarchical learning with variable population. *IEEE Trans. Syst. Man Cybern. Syst.* **2023**, *53*, 1397–1407. [CrossRef]
44. Xu, S.H.; Liu, J.P.; Zhang, F.H.; Wang, L.; Sun, L.J. A combination of genetic algorithm and particle swarm optimization for vehicle routing problem with time windows. *Sensors* **2015**, *15*, 21033–21053. [CrossRef] [PubMed]
45. Vidal, T.; Crainic, T.G.; Gendreau, M.; Prins, C. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Comput. Oper. Res.* **2013**, *40*, 475–489. [CrossRef]
46. Cui, H.X.; Qiu, J.L.; Cao, J.D.; Guo, M.; Chen, X.Y.; Gorbachev, S. Route optimization in township logistics distribution considering customer satisfaction based on adaptive genetic algorithm. *Math. Comput. Simul.* **2023**, *204*, 28–42. [CrossRef]
47. Wang, Y.Y.; Zhang, R.X.; Liu, H.; Zhang, X.Q.; Liu, Z.W. Task scheduling model of double-deep multi-tier shuttle system. *Processes* **2019**, *7*, 604. [CrossRef]
48. Chen, X.; Tianfield, H.; Mei, C.L.; Du, W.L.; Liu, G.H. Biogeography-based learning particle swarm optimization. *Soft Comput.* **2017**, *21*, 7519–7541. [CrossRef]
49. Bhattacharjee, K.K.; Sarmah, S.P. Shuffled frog leaping algorithm and its application to 0/1 knapsack problem Kaushik Kumar. *Appl. Soft Comput.* **2014**, *19*, 252–263. [CrossRef]
50. Nadimi-Shahraki, M.H.; Moeini, E.; Taghian, S.; Mirjalili, S. Discrete improved grey wolf optimizer for community detection. *J. Bionic Eng.* **2023**, *20*, 2331–2358. [CrossRef]
51. Wang, S.H.; Li, X.Y.; Liu, Q.H. An effective neighborhood solution clipping method for large-scale job shop scheduling problem. *CMES-Comput. Model. Eng. Sci.* **2023**, *137*, 1871–1890. [CrossRef]


**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.