

Article

Scheduling of Collaborative Vegetable Harvesters and Harvest-Aid Vehicles on Farms

Xiao Han ^{1,2,3}, Huarui Wu ^{1,2,3,*}, Huaji Zhu ^{1,2,3}, Jingqiu Gu ^{1,2,3}, Wei Guo ^{1,2,3}  and Yisheng Miao ^{1,2,3}

¹ National Engineering Research Center for Information Technology in Agriculture, Beijing 100097, China; hanx007@nercita.org.cn (X.H.); zhuhj007@nercita.org.cn (H.Z.); gujq007@nercita.org.cn (J.G.); guowei007@nercita.org.cn (W.G.); miaoys007@nercita.org.cn (Y.M.)

² Beijing Research Center for Information Technology in Agriculture, Beijing Academy of Agriculture and Forestry Sciences, Beijing 100097, China

³ Laboratory of Digital Village Technology, Ministry of Agriculture and Rural Affairs, Beijing 100097, China

* Correspondence: wuhr007@nercita.org.cn

Abstract: Transporting harvested vegetables in the field or greenhouse is labor-intensive. The utilization of small harvest-aid vehicles can reduce non-productive time for farmers and improve harvest efficiency. This paper models the process of harvesting vegetables in response to non-productive waiting delays caused by the scheduling of harvest-aid vehicles. Taking into consideration harvesting speed, harvest-aid vehicle capacity, and scheduling conflicts, a harvest-aid vehicle scheduling model is constructed to minimize non-production waiting time and coordination costs. Subsequently, to meet the collaborative needs of harvesters, this paper develops a discrete multi-objective Jaya optimization algorithm (DMO-Jaya), which combines an opposition-based learning mechanism and a long-term memory library to obtain scheduling schemes suitable for agricultural environments. Experiments show that the studied model can schedule harvest-aid vehicles without conflicts. Compared to the NSGA-II algorithm and the MMOPSO, the DMO-Jaya algorithm demonstrates a better diversity of solutions, resulting in a shorter non-productive waiting time for harvesters. This research provides a reference model for improving the efficiency of vegetable harvesting and transportation.

Keywords: vegetables in the field/greenhouse; non-productive waiting; collaborative of harvesting and transportation; discrete multi-objective optimization



Citation: Han, X.; Wu, H.; Zhu, H.; Gu, J.; Guo, W.; Miao, Y. Scheduling of Collaborative Vegetable Harvesters and Harvest-Aid Vehicles on Farms. *Agriculture* **2024**, *14*, 1600. <https://doi.org/10.3390/agriculture14091600>

Academic Editor: John M. Fielke

Received: 22 July 2024

Revised: 7 September 2024

Accepted: 11 September 2024

Published: 13 September 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

China, a significant consumer and producer of vegetables [1], faces complexity in the vegetable production process, with a limited availability of intelligent machines suitable for this task [2,3]. Manual harvesting, in particular, consumes a large amount of labor [4]. Farmers need to transport vegetables to nearby collection stations, which currently rely on collaborative work between harvesters and harvest-aid vehicles. While large agricultural technology companies have developed vegetable harvesters for specific crops, such as chili peppers, kale, and carrots, their application on a large scale remains limited due to high costs. In this context, scientific research institutions, such as the Beijing Academy of Agricultural and Forestry Sciences, have studied intelligent harvest-aid vehicles equipped with weight sensors and GPS modules.

The cost of an intelligent harvest-aid vehicle is significantly higher compared to that of a regular loading vehicle. On large farms, there can be numerous crops that require harvesting during the harvest season. Equipping each harvester with multiple harvest-aid vehicles for harvesting is not feasible. Sequential crop harvesting could involve multiple harvest-aid vehicles for each harvester, but this approach may cause delays in crop harvesting and result in losses. Therefore, it is necessary to study the collaborative scheduling of harvest-aid vehicles and harvesting operations. These vehicles offer flexibility, compactness, and improved efficiency by reducing non-productive labor time for farmers.

However, the scheduling of harvest-aid vehicles still relies on manual processes, leading to challenges such as empty running and blocking during large-scale harvesting tasks [5]. Unlike other industrial scenarios, agricultural environments are influenced by agronomic factors and crop conditions, requiring further exploration of digital modeling techniques [6]. Furthermore, there is a lack of standardization in the driving path and collaborative manner of vegetable harvesting and transportation.

To address these issues, this paper proposes a cooperative method for vegetable harvesting and transportation, considering factors such as harvester speed, efficiency, driving path, and load capacity. The objective is to minimize the waiting time for harvesters, ensure the conflict-free scheduling of harvest-aid vehicles, and optimize the efficiency of harvester collaboration [7]. By studying the collaboration between harvest-aid vehicles and harvesters (CHVH), this research aims to provide reference solutions for improving vegetable harvesting efficiency and promoting intelligent development in vegetable production.

In the field of agriculture, the cooperative scheduling of agricultural machinery has primarily focused on food crops [8,9]. Liu et al. addressed the issue of low efficiency in the continuous multitasking of agricultural machinery in hilly regions by proposing a scheduling model with the objective of minimizing the total scheduling time. To tackle this problem, they introduced an improved version of the non-dominated sorting genetic algorithm III [8]. Kan et al. proposed a random task allocation method, known as Next-Best Action Planning, specifically for grape picking. This approach captured the action constraints of agricultural machinery in an agricultural environment through SAG. By considering resource and energy budgets, the algorithm simultaneously determined the best sampling location and the optimal time to return to the collection station [10,11].

Additionally, Peng et al. focused on the dynamic scheduling of multiple robots in strawberry harvesting. They established a transportation function that comprehensively considered vehicle speed, unloading time, and harvesting time [12]. Their modeling efforts led to a near-optimal scheme for handling requests, providing an upper-efficiency limit for scheduling algorithms. Furthermore, Yang et al. integrated agricultural machinery resources using blockchain technology and established matching functions based on factors such as weather, road conditions, cost, and benefit. They employed genetic algorithms to supervise and schedule agricultural machinery [13]. In terms of harvesting, the failure rate of agricultural machinery tends to increase significantly. To address the uncertain needs of machinery maintenance services, Hu et al. proposed a scheduling model that minimized scheduling distance and the fixed costs of scheduled agricultural machinery [14]. Pan et al. proposed a deep reinforcement-learning-based method for scheduling multiple agricultural machineries, aiming to minimize crop losses and completion time in emergency scenarios. The experimental results demonstrated that this method outperformed existing approaches, with an average improvement of 26.7% in computation rate and 21.9% in completion time, while also significantly enhancing computational efficiency [15].

Large-scale autonomous robot networks have found widespread applications in logistics and various industries, including logistics warehouses [16–18], unmanned container terminals [19], and intelligent transportation systems [20–22]. As a notable example, Jing-Dong, a Chinese enterprise, operates the “Asia No.1” unmanned warehouse, which is a fully automated facility that handles the entire process of harvesting, storage, packaging, and order selection. To optimize robot scheduling in intelligent warehouses, Ma et al. considered the part segmentation caused by constraints such as part arrival time and mobile robot energy. They conducted a study using a variable neighborhood-search-based nondominated sorting genetic algorithm to obtain Pareto solutions that maximized chemical station satisfaction and minimized energy consumption. This approach enabled the generation of multiple high-quality mobile robot scheduling schemes, leading to improved workstation satisfaction and reduced energy consumption [23]. In a similar vein, Liu et al. proposed a multi-objective task allocation function that simultaneously optimized time cost and potential path conflicts for large-scale autonomous robot networks. They utilized a generalized conflict graph to transform the multi-objective function into a linear pro-

gramming example, resulting in reduced robot congestion and improved system operation efficiency [24]. Furthermore, Ham studied the real-time scheduling of production stations and transfer robots in flexible workshops, employing constraint programming to minimize the maximum completion time of tasks in FJSP+ robot scheduling [25]. Gultekin et al. proposed second-order cone programming and a heuristic algorithm to find Pareto solutions based on factors such as robot order, movement speed, energy consumption, and system production speed. This approach allowed for the generation of a robot scheduling plan with a small number of machines within a reasonable computational time [26]. Additionally, a novel middleware was proposed for compiling and building multi-robot collaborative applications [27].

The research on agricultural machinery scheduling primarily centers around resource allocation and path planning for agricultural machinery in the field, with a specific focus on harvesting grain crops or strawberries. Furthermore, the constraints and objective functions associated with scheduling multiple machines in an industrial context differ significantly from those in agriculture. Consequently, directly applying existing methods to the agricultural setting poses challenges. The core issue in this paper is how to integrate vegetable planting agronomy into the development of a collaborative scheduling model for multiple harvesters and harvest-aid vehicles in dynamic environments. While multiple robot scheduling has been successfully applied in industrial production, theoretical studies can be further improved based on actual production effects [28]. In the agricultural field, research on the cooperative operation of agricultural machinery is relatively recent, with limited consideration given to different crops, various agricultural machineries, and diverse agronomic needs. However, some studies have focused on cooperative operation scheduling for harvesting, primarily centered around static scheduling. Once certain conditions are met, cooperative operation requests are issued, and harvest-aid vehicles are directed by the decision-making center to predetermined locations. It is important to recognize that both the harvester and harvest-aid vehicles are dynamic during the harvesting process, and factors such as planting agronomics and machine size introduce constraints to the path and arrival time. Therefore, this paper aims to address the cooperative scheduling of agricultural machinery in the context of vegetable harvesting and transportation.

2. Developing a Collaborative Model for Vegetable Harvesters and Harvest-Aid Vehicles

2.1. Problem Description

To ensure efficient scheduling and minimize non-operation waiting time, a collaborative model is designed for harvest-aid vehicles to work in conjunction with vegetable harvesting and transportation. This model takes into account the transportation needs of different harvesters and aims to optimize the scheduling efficiency of these harvest-aid vehicles. By integrating a communication module, path planning module, and weighing sensor, each harvest-aid vehicle can engage in real-time interactive communication with the central decision-making system and the harvester. Similarly, the harvester is equipped with a communication module and autonomous path planning module to facilitate interaction with the central decision-making system and harvest-aid vehicles. Utilizing data collected by weight sensors, the vehicle can determine if it is nearing full capacity and request the scheduling of an empty vehicle if needed. If no available vehicle is present, the harvester will stop working and wait for further scheduling instructions.

As shown in Figure 1, vegetables in the field are typically grown on furrows with a width ranging from 55 to 75 cm. However, the unique planting structure of field vegetables poses challenges to the collaboration between harvesters and harvest-aid vehicles. The width of agricultural machines, such as leafy vegetable harvesters like the Assali, is much larger than the width of furrows. In most cases, harvesters operate in a backhoe mode. Because farms are open environments, most roads must allow large agricultural machinery to pass through. Roads on farmland are relatively wide, and non-operational path conflicts can be disregarded. These paths facilitate bidirectional driving for different vehicles. Scheduled vehicles enter the field through fixed entrances, while for greenhouses, vehicles only

load vegetables at the greenhouse entrance. Each harvest-aid vehicle transports a portion of the harvested vegetables to a specific collection station, independently determining which station to return to.

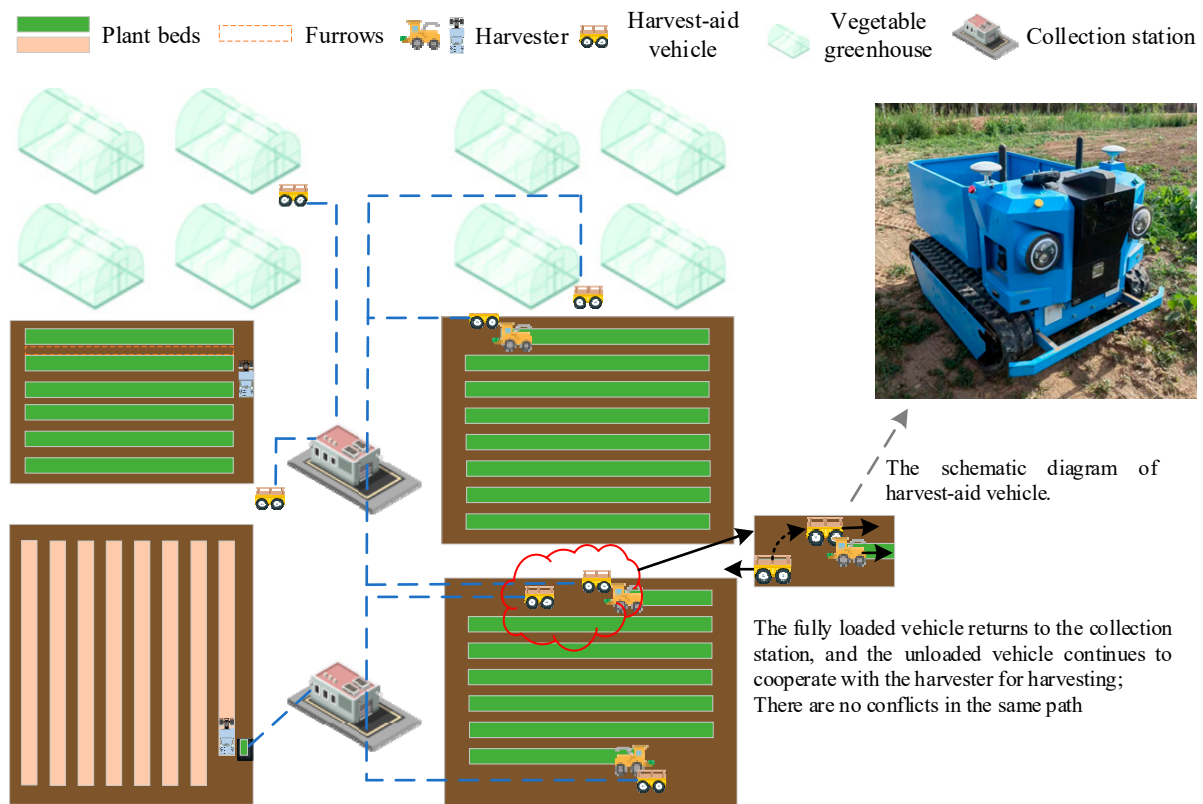


Figure 1. Harvest-aid vehicle scheduling for harvesting and transportation collaboration.

The collaborative harvesting and transportation process can be described as follows: On a farm, the number of vegetable fields/greenhouses to be harvested is denoted as V , the number of harvesters as P , and the number of harvest-aid vehicles as N . The harvest-aid vehicles are required to unload at the collection station. Multiple harvest-aid vehicles can alternately complete the harvesting of vegetables in the same block, with each harvester being followed by only one vehicle at a time. The goal is to select and schedule the appropriate harvest-aid vehicle for each harvester in different blocks, optimizing the overall scheduling time and minimizing the non-operating waiting time. To simplify the research scenario, the following assumptions are made based on agronomic needs and the operation status of harvesters:

1. All harvest-aid vehicles start from the collection station, pick up vegetables, transport them to the collection station when fully loaded, and then proceed to their designated position to assist the harvester in vegetable harvesting according to the scheduling scheme determined by the decision center.
2. The harvest-aid vehicles adhere to a fixed unloading sequence at the collection station, with a predetermined unloading time.
3. Not all harvest-aid vehicles start from the collection station; some may still be on route, having not yet reached their loading threshold. These vehicles also need to be factored into the scheduling process.
4. Each harvester is supported by only one harvest-aid vehicle at a time.
5. Each harvest-aid vehicle departs from the collection station upon receiving a scheduling request and arrives at its designated position to collect vegetables until the weight threshold is reached. The transportation of vegetables to the collection station is a fully collaborative process.

6. Each field is a regular rectangle, and both the harvest-aid vehicle and harvester enter the field through a fixed entrance. The variables involved in vegetable harvesting and transportation are presented in Table 1.

Table 1. The variables involved in vegetable harvesting and transportation.

Parameters	Parameter Meaning	Parameters	Parameter Meaning
$Tk = 1, \dots, i, \dots, N$ $i, 1 \leq i \leq N$	Harvest-aid vehicle set Harvest-aid vehicle ID	W_{max}	The full load weight of harvest-aid vehicle
$Tr = 1, \dots, j, \dots, P$ $j, 1 \leq j \leq P$	Harvester set Harvester ID	vt_i	The driving speed of harvest-aid vehicle i when going to the field and returning to the collection station (m/min)
$Cl = 1, 2$ $l, l = 1, 2$	Collection station set Collection station ID	s_i	The distance between harvest-aid vehicle i and the collection station
$Gr = 1, \dots, r, \dots, R$ $r, 1 \leq r \leq R$	Block Set Block ID	sg_{rl}	The distance between the entrance to the block r and the collection station l
Se_i	Unloading sequence of harvest aid vehicle i at the collection station	st_{ij}	The distance between the harvest-aid vehicle i located at the collection station and the expected position
t	The time when the scheduling request is issued	sk_{ij}	The distance between the harvest-aid vehicle i located at the block entrance and the harvester j
tc_i	The time when the vehicle i arrives at the expected location	p_j	The harvesting efficiency of harvester j (harvesting vegetable kg/min)
Δtw_i	The time interval required for unloading the harvest-aid vehicle i	θ	The driving direction of harvester
Δts_{ij}	The time interval between harvest aid i leaving the collection station and moving to the expected location of harvester j	ϑ	The furrow's heading direction
Δtt_i	The time interval for harvest-aid vehicle i to return to the collection station	vc_j	The driving speed of harvester j (m/min)
W_i	Load of harvest aid vehicle i	(tx_j, ty_j)	The location coordinates of harvester j at time t
W_{th}	Load threshold for harvest-aid vehicles	(cx_r, cy_r)	The location coordinates of entrance r

2.2. The Model of Harvester and Harvest-Aid Vehicle

During vegetable harvesting in the field, each harvester collects vegetables, accompanied by a harvest-aid vehicle that maintains the same driving speed. This vehicle follows the harvester j , collecting and transporting vegetables. When the current harvest-aid vehicle, denoted as vehicle i , is about to be fully loaded at time t , a new request Re is generated and sent to the decision center. Re includes the following information:

1. The remaining time interval Δtf : The time between the current harvest-aid vehicle being requested and the time it becomes fully loaded.
2. The predicted coordinates of the harvester at the time when the current harvest-aid vehicle becomes full.

When the harvest-aid vehicle, pr , is about to be fully loaded based on the weight sensor reading ($W_{pr} > W_{th}$), it sends a request Re to the decision-making center for replacement. At this point, the decision-making center can schedule another available harvest-aid vehicle to take its place. Firstly, it determines if the requesting vehicle can assist the harvester in completing the vegetable harvest on the current plant bed before being fully loaded. If not, the scheduling time for the harvest-aid vehicle is calculated using the following method. If it can, the expected position is set to the starting position of the next plant bed that the harvester will work on, which can be determined from the harvester's path planning system. The specific scheduling time calculation follows the same procedure as in the harvester's running state, which will not be discussed here. If the scheduled harvest-aid

vehicle can arrive at the expected location before the current vehicle is fully loaded, the corresponding harvester does not need to stop and wait. The scheduling cost consists of the time Δtt_{il} for the harvest-aid vehicle i to return to the collection station (if it is currently located there, $\Delta tt_{il} = 0$), the time Δtw_i to unload the collected vegetables, and the time Δts_{ij} to drive to the expected location.

$$\Delta tw_i = \begin{cases} se_i \times t_{uni} + tw_0, & se_i = 1, 2, \dots, Ns \\ tw_0, & se_i = 0 \end{cases} \quad (1)$$

$$\Delta tt_i = \frac{s_i}{vt_i} \quad (2)$$

$$\Delta ts_{ij} = \frac{st_{ij}}{vt_i} \quad (3)$$

When the unloading priority of harvest-aid vehicle i is $se_i = 0$, it indicates that the vehicle is in an empty state, and the remaining unloading time tw_0 is 0.

The distance st_{ij} is calculated as follows:

$$st_{ij} = sg_{rl} + sk_{ij} \quad (4)$$

In Equation (4), sg_{rl} represents the distance between the collection station r (where the scheduled harvest-aid vehicle i is located) and the entrance to the block l (where the harvester j is located) at the time of the scheduling request. It can be observed that sg_{rl} is fixed. However, since the harvester is dynamic, sk_{ij} should be calculated based on the advanced distance and operation mode of harvester j . Taking into account the characteristics of vegetable planting, this paper analyzes and models sk_{ij} .

(1) As illustrated in Figure 2a, when the entrance is perpendicular to the furrow, the scheduled vehicle i first arrives at the head of the furrow where the harvester is located from the entrance. This distance is denoted as $d1_{ij}$, which can be easily calculated. Then, the distance between the transport vehicle from the furrow head to the location (tx_j, ty_j) of the harvester at time t is $d2_{ij}$. Based on the scheduling time and the driving speed of the harvester, the expected distance $d3_{ij}$ can be calculated. The distance from the entrance to the expected location is determined accordingly. When the block distribution is represented as shown in Figure 2, $d3_{ij}$ can be obtained using the following equation:

$$d3_{ij} = (\Delta tw_i + \Delta tt_i + \Delta ts_{ij})vc_j \sin(\theta_j - \theta_r) \quad (5)$$

(2) As depicted in Figure 2b,d, when there are unharvested vegetables between the harvester awaiting assistance and the entrance, the scheduled harvest-aid vehicle first arrives at the furrow head of the nearest furrow along the furrow direction. This distance is denoted as $d1_{ij}$. Then, the vehicle drives in a direction perpendicular to the furrow heading to reach the furrow where the harvester needs to be coordinated, with this distance set as $d2_{ij}$. The distance between the transport vehicle from the furrow head to the location (tx_j, ty_j) of the harvester at time t is $d3_{ij}$. Calculating $d1_{ij}$, $d2_{ij}$, and $d3_{ij}$ is straightforward. Lastly, the distance between the harvest-aid vehicle and the location to be coordinated is set as $d4_{ij}$, which can be calculated similarly to Equation (5). In this scenario, the distance from the entrance to the expected position of the harvest-aid vehicle i is $sk_{ij} = d1_{ij} + d2_{ij} + d3_{ij} + d4_{ij}$. By deriving the expected position and Euclidean distance from $d1_{ij}$, $d2_{ij}$, and $d3_{ij}$, sk_{ij} can be obtained.

The decision-making center performs a comprehensive calculation to determine whether the current harvest-aid vehicle can harvest all the vegetables on the current furrow before it becomes fully loaded. If this is the case, the expected position of the harvest-aid vehicle i , corresponding to the scheduling, will be the same as the starting position of the next furrow that the harvester expects to harvest.

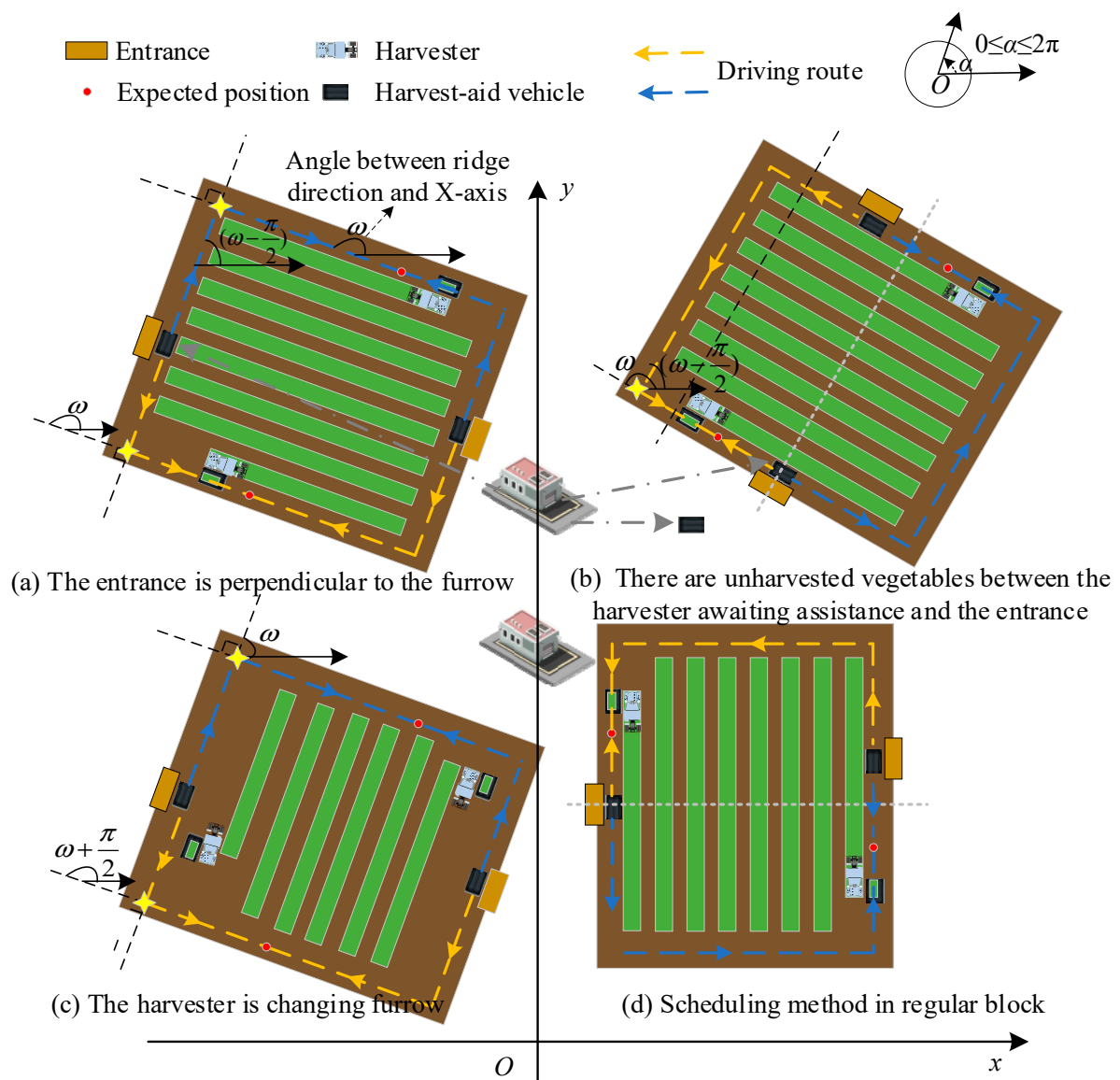


Figure 2. Harvest-aid vehicle scheduling diagram with different block distribution.

However, if the schedulable harvest-aid vehicle i cannot reach the expected location before the harvest-aid vehicle to be replaced is fully loaded, the corresponding harvester needs to stop and wait for the scheduled vehicle. The waiting time of the harvester j is defined as:

$$t_{wait_j} = \max(\Delta tw_i + \Delta tt_i + \Delta ts_{ij} - \frac{W_{max} - W_{pr}}{p_j}, 0) \quad (6)$$

When there are no available harvest-aid vehicles, the system waits for these vehicles to become idle and then selects the scheduling scheme with the lowest scheduling cost, as determined by Equation (7). During vegetable harvesting in a greenhouse, the harvest-aid vehicle loads vegetables at the greenhouse entrance, and the scheduling distance is calculated using the Euclidean distance. The scheduling model for harvest-aid vehicles should prioritize avoiding frequent stops by harvesters, as this can incur rental expenses and productivity losses due to frequent start-stop operations. Taking into consideration the scheduling cost of harvest-aid vehicles and harvesting efficiency, the following scheduling model is constructed:

Objection :

$$f_1 = \min \sum_{j=1}^P (1 - h_j) t_{wait_j}$$

$$f_2 = \min \sum_{i=1}^N \sum_{j=1}^P ((\Delta tw_i + \Delta tt_i + \Delta ts_{ij}) \times z_{ij})$$

s.t.

$$C1 \quad 0 \leq W_{pr} \leq W_{max}, \forall i \in Tk$$

$$C2 \quad pr \neq i, \forall pr, i \in Tk$$

$$C3 \quad \sum_{i=1}^N z_{ij} \leq 1, \forall j \in Tr$$

$$C4 \quad \sum_{j=1}^P z_{ij} \leq 1, \forall i \in Tk$$

$$C5 \quad tc_i \geq t + \Delta tt_i + \Delta tw_i$$

$$C6 \quad z_{ij} = \{0, 1\}, \forall i \in Tk, \forall j \in Tr$$

$$C7 \quad h_j = \{0, 1\}, \forall j \in Tr$$

$$C8 \quad [t, t + \Delta tt_i + \Delta tw_i) \cup [tc_i - \Delta ts_{ij}, tc_i) = \phi \forall i \in Tk, \forall j \in Tr$$

(7)

The objective function of the scheduling model is composed of two factors: f_1 , which minimizes the waiting time of all operational harvesters, and f_2 , which minimizes the scheduling cost of harvest-aid vehicles while ensuring the continuous operation of the harvester to be assisted. Additionally, there are constraints to consider. C1 ensures that the load of the harvest-aid vehicle does not exceed the maximum load capacity. C2 ensures that the harvest-aid vehicle that issued the request is not the same as the vehicle that responded to the schedule. C3 indicates that each vehicle can currently only collaborate with one harvester. C4 states that each harvester is coordinated by only one harvest-aid vehicle simultaneously. C5 specifies that the harvest-aid vehicle can only be scheduled after unloading. The decision variables are z_{ij} and h_j . Finally, C8 ensures that the vehicle returns to the collection station and completes unloading before heading to the expected location.

3. Discrete Multi-Objective Jaya Algorithm

In 2016, Rao proposed the Jaya algorithm as an efficient and easy-to-run method for solving complex optimization problems [29]. This algorithm requires only a small number of control parameters, namely the predefined population size and the maximum number of iterations. During the optimization process, X_{mn} represents the value of the n -th dimensional variable of the m -th solution in the population. X_{best} and X_{worst} denote the optimal and worst solutions in the population, respectively. The Jaya algorithm utilizes two random numbers, $ra_{1,j}$ and $ra_{2,j}$, generated in the j -th dimensional space, where $ra_{1,j}$ and $ra_{2,j} \in [0, 1]$. $ra_{1,j}$ and $ra_{2,j}$ are random numbers within the range of $[0, 1]$, introducing randomness to the algorithm to help it escape local optima and enhance its global search.

The Jaya algorithm starts by forming a random initial population, and then iteratively adjusts the individuals to approach the optimal solution and move away from the worst solution, aiming to find the best solution. During the iteration process, the algorithm dynamically adjusts the positions of candidate solutions based on the performance of the current population, so there is no need for additional control parameters to adjust the search strategy of the algorithm. Compared to some other optimization algorithms, the performance of the Jaya algorithm is usually less sensitive to parameters. The update method is as follows [30]:

$$X'_{mn} = X_{mn} + ra_{1,j}(X_{best} - |X_{mn}|) - ra_{2,j}(X_{worst} - |X_{mn}|) \quad (8)$$

In the traditional Jaya algorithm, if the $F(X'_{mn})$ corresponding to the solution X'_{mn} is better, then X'_{mn} is accepted and replaces the original solution; otherwise, the original solution is maintained. The term $ra_{1,j}(X_{best} - |X_{mn}|)$ indicates that the solution is close

to the optimal solution, and $-ra_{2,j}(X_{i_{worst}} - |X_{min}|)$ suggests that the solution is far from the worst solution. It is worth noting that the traditional Jaya algorithm is primarily used for addressing continuous single-objective optimization functions. To adapt it for discrete harvest-aid vehicle scheduling models, this paper introduces a discrete multi-objective Jaya algorithm. The main steps of this algorithm include encoding, decoding, initialization, individual updating, and Pareto sorting.

(1) Encoding and decoding

Firstly, the solutions for collaborative requests and harvest-aid vehicle scheduling are encoded as shown in Figure 3. Each solution is represented by a matrix that contains the scheduled harvest-aid vehicle ID. The encoding length is equal to the total number of vehicles. For example, if the code for collaborative requests is [1, 5, 4, 2, 6, 3], and $X = [2, 3, 1, 4, 5]$, it indicates that harvester 1 is assisted by harvest-aid vehicle 2, harvester 5 is assisted by harvest-aid vehicle 3, and so on.

Cooperative request of harvester

1	5	4	2	6	3
2	3	1	4	5	6

Harvest-aid vehicle ID

Figure 3. Schematic diagram of scheduling scheme coding.

Next, when a harvest-aid vehicle arrives at the location of harvester j before it stops and waits, the latest harvest-aid vehicle is selected. On the other hand, if the vehicle arrives after the harvester has stopped, the earliest harvest-aid vehicle that reaches the location of harvester j is chosen. By following this scheduling rule to respond to cooperative transport requests, the maximum utilization efficiency of harvest-aid vehicles can be achieved, and the coordination time can be minimized.

Decoding is the process of determining the minimum scheduling cost and generating a specific coordination scheme for harvesting and transportation based on the encoding. In the decoding phase, when a vehicle completes a round of vegetable harvesting and unloading, it becomes idle at the collection station. If the next task for the harvest-aid vehicle is to assist harvester j , the vehicle needs to reach the expected position.

(2) Pareto optimization

In multi-objective optimization, it becomes challenging to determine the optimal and worst solutions, since there are multiple objectives to consider. To effectively address this, a combination of non-dominated ranking and a crowding mechanism is used for Pareto optimization. The solution group can be ranked based on the dominance theory. The rule is as follows:

For two solution schemes, X_e and X_q , if X_e is not inferior to X_q in all objectives and is strictly superior to X_q in at least one objective, then X_e dominates X_q , indicating that X_e has a higher rank. The solution with a higher rank is considered to be superior to other solutions. In the case of two solutions with the same rank, the solution with a higher crowding distance is considered to be superior to the others.

$$CD_q = \sum_{k=1}^K \frac{f_k^{q+1} - f_k^{q-1}}{f_k^{\max} - f_k^{\min}} \quad (9)$$

where K is the number of objective functions, q is the j -th particle in the population, and f_k^{q+1} and f_k^{q-1} represent the k -th objective function values of the two positions adjacent to q . Furthermore, $f_k^{\max}$ and $f_k^{\min}$ represent the maximum and minimum values of the k -th objective function in the population, respectively. The optimal solution is determined based on the highest level and the highest degree of crowding, while the worst solution is

determined accordingly. Once these two solutions are identified, the individuals in the new population will be updated.

(3) Pareto optimization

For individual updates, the existing Jaya algorithm's individual update method is suitable for continuous variables and cannot be directly applied to discrete optimization problems. To effectively update individuals, specific update methods are designed for different situations:

- (a) If X_{mn} is identical to X_{worst} or X_{best} , a new individual X'_{mn} is regenerated.
- (b) If X_{mn} is not exactly the same as X_{worst} or X_{best} , based on the principles of the Jaya algorithm, the code in X_{mn} that matches X_{worst} is deleted and replaced with the code at the corresponding position in X_{best} , thereby generating a new individual X'_{mn} . However, considering the time constraints of vehicle scheduling, if the replacement code at the corresponding location in X_{best} is the same as another location code, the location code is regenerated. The individual update methods in this mode are illustrated in Figure 4.

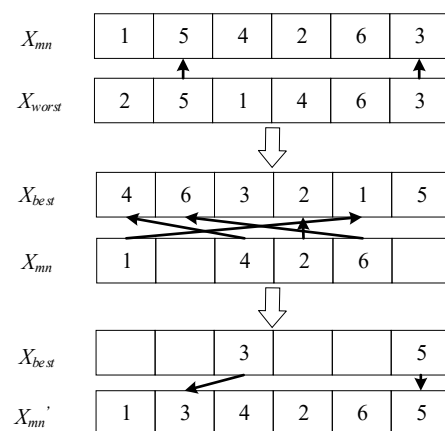


Figure 4. Individual updates.

The current optimal and worst solutions in the population play crucial roles in guiding individuals towards a global optimal solution. However, if these solutions fall into a local extreme region, this can result in search stagnation within the population and hinder the attainment of a better global optimal solution [31]. To address this issue, this paper explores the discrete Jaya algorithm based on an opposition-based learning mechanism. By incorporating opposition-based learning into the current optimal and worst solutions, the algorithm increases the likelihood of escaping from local extreme value regions. It is important to note that opposition-based learning is only performed on the current optimal and worst individuals, rather than all individuals individually [32]. This approach helps to prevent the Jaya algorithm from falling into local optimization while maintaining convergence speed.

Assuming that $X_{best}(iter)$ represents the individual of the $iter$ -th population, the opposition-based learning process is as follows:

$$X'_{best}(iter) = (1 + Nr)\eta - X_{best}(iter) \quad (10)$$

$$X_{best} = \begin{cases} X'_{best} & \text{if } f(X'_{best}) < f(X_{best}) \\ X_{best} & \text{else} \end{cases} \quad (11)$$

$$X'_{worst}(iter) = (1 + Nr)\eta - X_{worst}(iter) \quad (12)$$

$$X_{worst} = \begin{cases} X'_{worst} & \text{if } f(X'_{worst}) > f(X_{worst}) \\ X_{worst} & \text{else} \end{cases} \quad (13)$$

where Nr represents the total number of scheduling requests. X'_{best} and X'_{worst} represent the inverse individuals of the optimal solution and the worst solution, respectively. It is

important to note that some harvest-aid vehicles are already engaged in work and cannot be rescheduled. For the sake of simplifying the opposition-based learning process, this paper performs opposition-based learning based on the coding order in X_{best} , rather than opposition-based learning of the code itself. The process of opposition-based learning is illustrated in Figure 5.

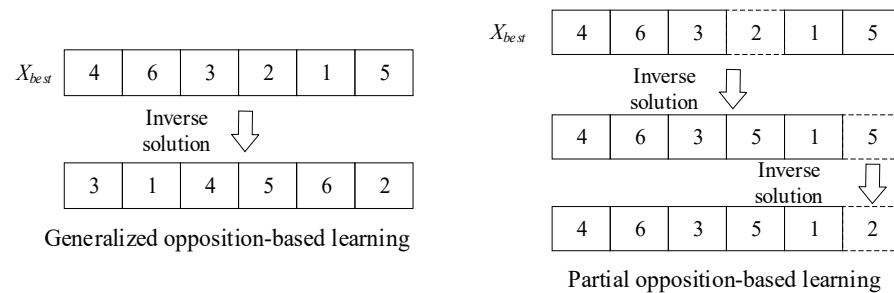


Figure 5. Schematic diagram of opposition-based learning.

The opposition-based learning mechanism described above primarily focuses on enhancing the local development ability of the algorithm. To further improve the solution accuracy and stability while enhancing the global exploration ability, long-term memory libraries are designed to increase the diversity of solutions [33]. This memory library establishes an elite solution memory with a size of H , denoted as $fl = \{fl_1, fl_2, \dots, fl_H\}$. The solution fl_H represents the H -th solution stored in the memory. X_m^* is a candidate solution obtained through iterative local search. To update the memory, X_m^* is compared with X_{m-H} , where X_{m-H} corresponds to fl_{imodH} in the memory. If X_m^* is superior, the objective function value of X_m^* replaces fl_{imodH} in the memory, and the current solution is replaced by X_m^* for the next iteration. The complete iteration process is illustrated by Algorithm 1.

$$X_m^* = \begin{cases} X_m^*, & \text{if } X_m^* \text{ dominate } X_{m-H} \\ X_m, & \text{else} \end{cases} \quad (14)$$

Algorithm 1 Discrete Multi-objective scheduling algorithm

- 1: Initialize parameters $Maxiter, D, Nr, H$
 - 2: Initialize population X
 - 3: Decode and compute the objective function $f_1(X)$
 - 4: Decode and compute the objective function $f_2(X)$
 - 5: According to Pareto sort, calculate the initial best solution X_{best} and the worst solution X_{worst}
 - 6: Set $t = 1$
 - 7: **while** $iter \leq Maxiter$ **do**
 - 8: According to the Equation (9), Pareto sort is used to select the current best solution X'_{best} and the worst solution X'_{worst}
 - 9: According to the opposition-based learning mechanism, obtain the inverse solution Xr'_{best} and Xr'_{worst} of X'_{best} and X'_{worst}
 - 10: **if** X'_{best} dominates X_{best} or X_{worst} dominates X'_{worst} **then**
 - 11: $X_{best} = X'_{best}$ or $X_{worst} = X'_{worst}$
 - 12: **else**
 - 13: $X_{best} = X_{best}$ or $X_{worst} = X_{worst}$
 - 14: **end if**
 - 15: **if** X_{best} dominates X_{iter-H} **then**
 - 16: $X_{iter-H} \leftarrow X_{best}$
 - 17: **else**
 - 18: $X_{best} \leftarrow X_{iter-H}$
 - 19: **end if**
 - 20: Renew individuals and generate new population according to Section 3
 - 21: **end while**
 - 22: Output Pareto frontier if the termination condition is met, otherwise go to 7
-

4. Experiment and Analysis

To validate the effectiveness of the collaborative scheduling approach proposed in this paper, experiments were conducted using a realistic scenario. Simulation verification was performed for the multi-harvest-aid vehicle scheduling. An overall farm was divided into different vegetable fields/greenhouses, as shown in Figure 6, with each workplace having its own entrance. Two collection stations were located at the center of the farm, where multiple harvest-aid vehicles and harvesters collaborated in the harvesting process. The vegetables in the fields were planted on beds, ranging in length from 100 to 250 m. Harvesting was performed in a backhoe manner, with the furrow heading direction being randomly generated within the range of $(0, 180^\circ]$.

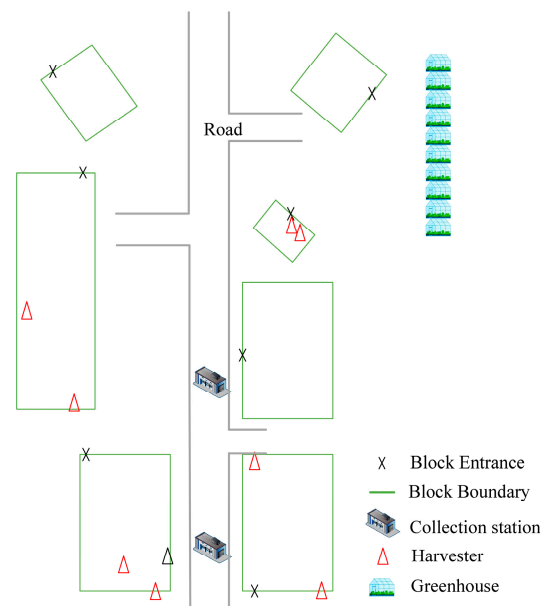


Figure 6. Schematic diagram of vegetable harvesting environment.

To evaluate the effectiveness of the method proposed in this paper, three examples were designed based on actual farm data. From each group of data, z harvest-aid vehicles were randomly selected from the agricultural machinery station as the initial idle harvest-aid vehicles currently owned by the decision-making center, and their locations and unloading sequences were randomly generated. At the same time, p harvest-aid vehicles that were about to be fully loaded were randomly generated, and the information for each vehicle included the harvester it was currently assisting, the current load, and whether it needed to be replaced. The farm consisted of g vegetable blocks and s vegetable greenhouses. Considering the different harvesting efficiencies of different vegetables, the task parameters are summarized in Table 2.

Table 2. Harvesting parameters.

	Harvesting Efficiency (kg/s)	Driving Speed (km/h)
Cabbage Harvesting	0.588	1
Pepper Harvesting	0.423	1.5
White Radish Harvesting	0.697	0.8

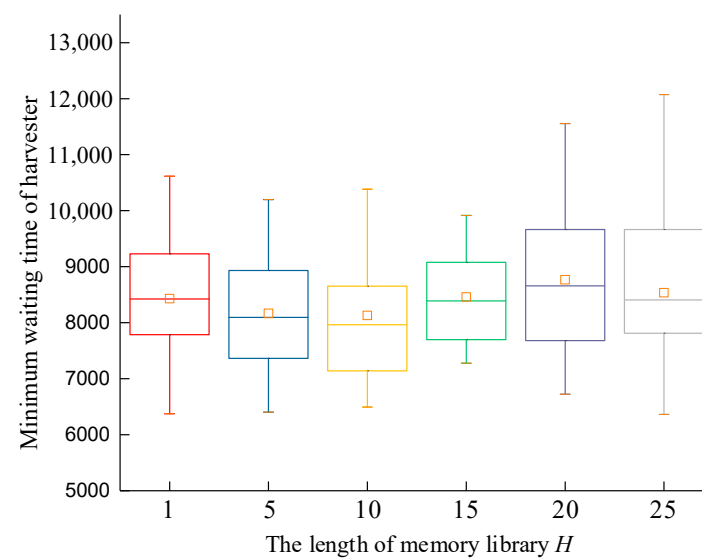
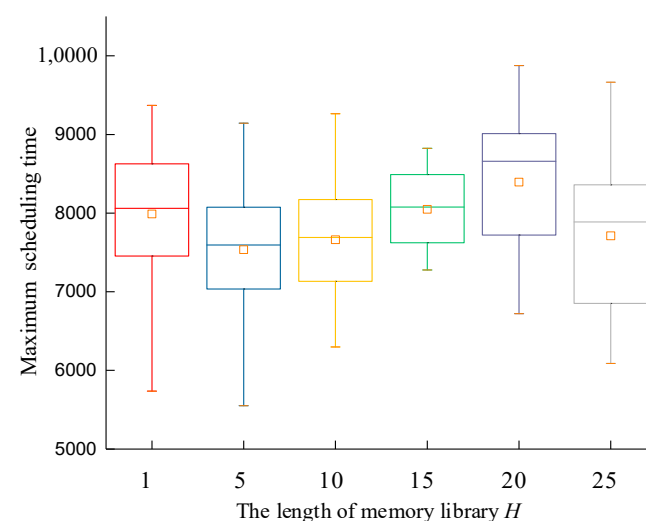
Additionally, the relevant parameters of the harvest-aid vehicles are presented in Table 3. It is worth noting that all the harvest-aid vehicles had the same structure.

Table 3. Harvest-aid vehicle parameters.

Parameter Description	Value
Non-production driving speed of the harvest-aid vehicle	2.5 km/h
Maximum load	400 kg
Load threshold	300 kg
Unloading time of a single harvest-aid vehicle	1 min

4.1. Algorithm Performance Verification

The parameter settings have a significant impact on the performance of the algorithm. Let $g = 7$, $s = 5$, $p = 6$, and $z = 10$. The maximum iteration times of the algorithm were set as $itermax = 1000$, the population size was set as $pop = 80$, and the optimal solution adopted the generalized opposition-based learning approach with a memory library $H = \{1, 5, 10, 15, 20, 25\}$. The experiment was repeated 30 times with the same parameters and examples. The experimental environment used an Intel (R) Core (TM) i7-9700H processor with 16 GB of memory. The average value of objective function f_1 and the average value of objective function f_2 are shown in Figures 7 and 8.

**Figure 7.** Box diagram of minimum waiting time distribution of harvesters with different H .**Figure 8.** Box type diagram of maximum scheduling time of harvest-aid vehicles with different H .

Each group of data were subjected to an S-W normal distribution test. It can be concluded that all 30 experimental results obtained at $H = \{1, 10, 15, 20, 25\}$ followed a normal distribution. The experimental results had a certain degree of stability, except when $H = 10$. Figures 7 and 8 show that, under different H lengths, the 30 repeated experiments showed that the average maximum waiting time of the harvester and the scheduling time were smaller when $H = 15$, and the box type was narrower, indicating a more stable distribution. Therefore, this paper sets $H = 15$.

Partial opposition-based learning extends multiple solutions by making inverse changes to partial positions of the solution. On the other hand, generalized opposition-based learning expands diversity by performing opposition-based learning on each position of the solution. In order to verify the learning mechanism applicable to the collaboration model in this paper, both partial opposition-based learning and generalized opposition-based learning were implemented. As shown in Figure 9, when the algorithm applied partial opposition-based learning, the average maximum waiting time of the harvester and the scheduling time of the harvest-aid vehicle were minimized, and the data distribution remained stable. Therefore, this paper compares algorithm performance and analyzes collaborative scheduling schemes using partial opposition-based learning.

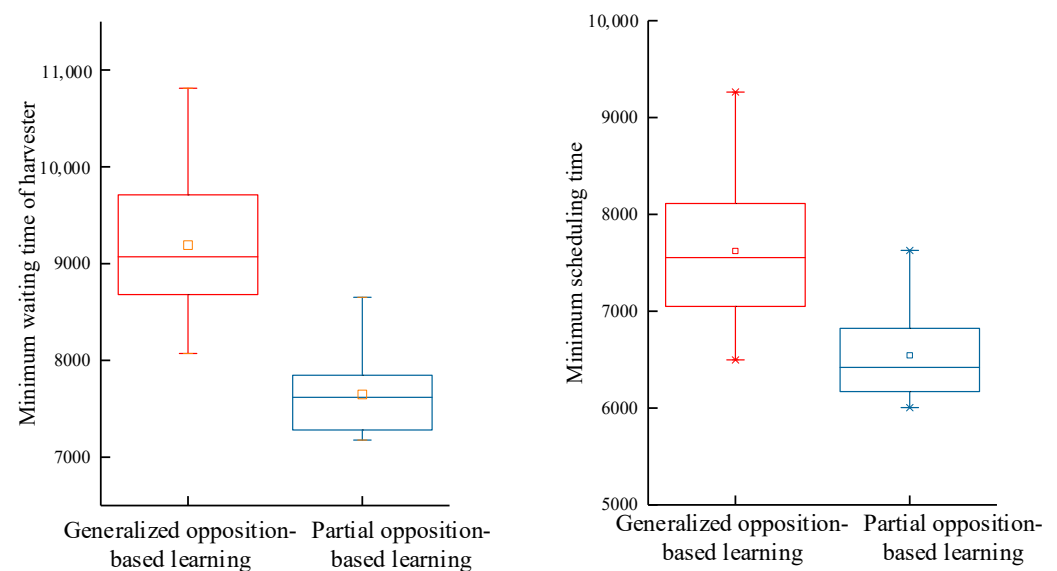


Figure 9. Comparison of algorithm performance with different opposition-based learning mechanisms.

4.2. The Comparison of Algorithm Performance

In the context of harvest-aid vehicle scheduling, this paper compares three commonly used algorithms as follows: NSGA-II (second-generation non-dominated sorting genetic algorithm), modified multi-objective particle swarm optimization (MMOPSO), and the traditional Jaya algorithm.

NSGA-II proposes a fast non-dominated sorting genetic algorithm that reduces computational complexity. It combines the parent population with the offspring population, allowing the next generation's population to be selected from a double space. This ensures that all the best individuals are preserved. By introducing elite strategies, NSGA-II prevents the discarding of excellent individuals during evolution, thereby improving optimization accuracy. Moreover, it utilizes crowding degree and crowding degree comparison operators, promoting the uniform expansion of individuals in the quasi-Pareto domain to the entire Pareto domain and ensuring population diversity.

On the other hand, the MMOPSO algorithm designs two search strategies based on particle swarm optimization to update the particle speed. It also shares elite information in external archives through an evolutionary search strategy that simulates binary crossover and polynomial mutation.

The convergence curves of the four algorithms with the same parameters are shown in Figure 10. Convergence means that, as the algorithm is executed, it will eventually reach a stable state or solution. This is the foundation for algorithms to obtain meaningful results. Figure 10a depicts the iterations of objective function 1 with different algorithms, while Figure 10b shows the iterations of objective function 2. It is evident that the collaboration between harvest-aid vehicles and harvesters (CHVH) exhibits a faster convergence speed and can obtain better solutions.

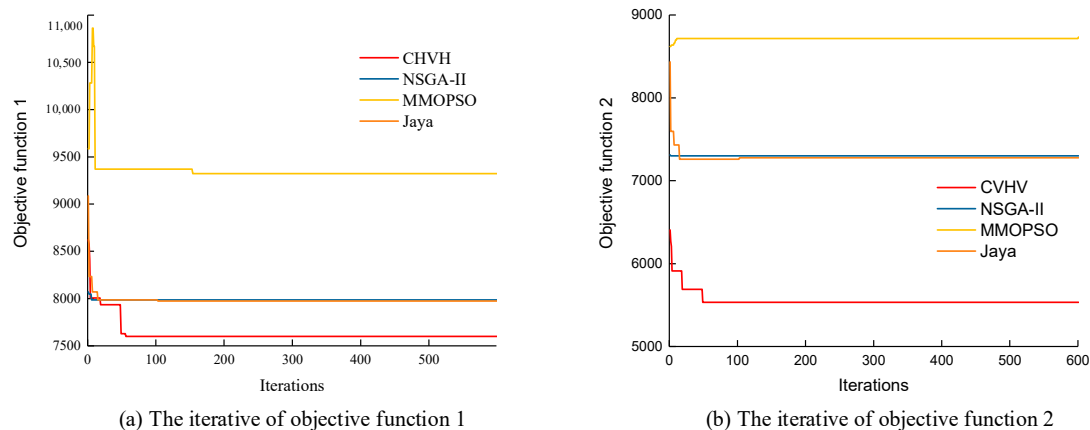


Figure 10. Convergence curves.

z represents the number of harvest-aid vehicles in the agricultural machinery station as the initial idle harvest-aid vehicles currently owned by the decision-making center. p represents the number of harvest-aid vehicles about to be fully loaded. g represents the number of vegetable blocks and s represents the number of vegetable greenhouses. In fact, all four parameters are determined by the scale and configuration of the farm. Table 4 shows a comparison between the HVH algorithm and other algorithms on farms of different sizes. AM represents the solution obtained by the CHVH algorithm. The scheduling efficiency was analyzed by calculating the gap between the solution obtained by other algorithms and AM. It can be seen that our CHVH algorithm was also superior to the other three commonly used heuristic algorithms. Among them, NSGA-II was superior to the MMOPSO algorithm, but there was still a gap between the CHVH algorithm and the multi-objective genetic algorithm. The algorithm in this paper had the shortest solution time and a higher efficiency. Reducing the non-productive waiting time of the harvester can more flexibly improve the scheduling efficiency and reduce the loss cost of the harvester. The discrete Jaya algorithm had a relatively short running time, but still lagged behind the CHVH algorithm in terms of the accuracy of the solution. The discrete multi-objective Jaya algorithm is of great significance in effectively reducing the operating costs of systems and achieving intelligent vegetable harvesting and transportation.

Table 4. Harvest-aid vehicle parameters.

Example	p	g	z	s	Compare the Total Distance Gap of AM						Running Time			
					NSGA-II		MMOPSO		Jaya					
Small-scale farm	7	7	15	10	5.1%	32.0%	28.2%	57.8%	4.9%	31.5%	10.0 s	11.9 s	108.6 s	9.34 s
	5	7	10	10	1.9%	1.9%	30.5%	26.4%	5.6%	12.0%	9 s	11.3 s	194.1 s	8.9 s
Large-scale farm	10	10	20	10	0.1%	0.1%	26.2%	23.7%	4.9%	10.3%	10.9 s	14.5 s	276.6 s	12.1 s
	10	10	25	20	0.7%	2.5%	31.5%	39.2%	5.3%	27.5%	11.4	13.4 s	126.4 s	10.8 s
Special scenarios	8	10	12	1	5.2%	4.2%	45.8%	45.2%	3.4%	27.5%	10 s	13.6 s	108.3 s	10.9 s
	6	1	12	35	1.1%	6.2%	25.0%	50.2%	0.1%	2.6%	9.3 s	11.5 s	25.6 s	8.9 s

$$\text{Gap} = (\text{Comparison} - \text{AM}) / \text{AM} \times 100\%.$$

To further analyze the distribution differences of the non-dominated solutions obtained by the CHVH algorithms, the non-dominated solutions were sorted after 1000 iterations. Consequently, the optimal non-dominated solution sets of the two algorithms were obtained, as shown in Figure 11. It can be seen that the Pareto front of the Jaya algorithm and the improved algorithm in this paper were clearly nonlinear. The solution obtained by CHVH demonstrated a wider distribution range and better diversity. Additionally, the scheduling scheme outperformed the comparison algorithms.

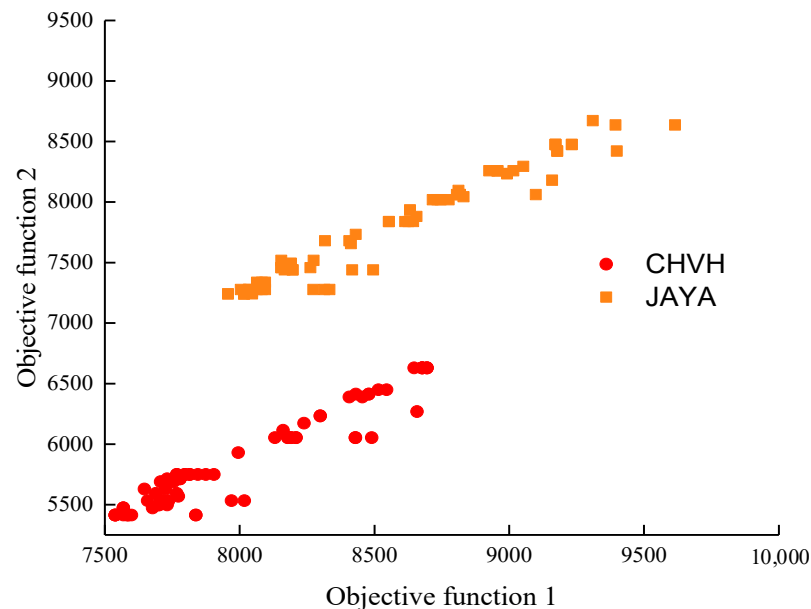


Figure 11. Pareto front.

5. Discussion

This paper addresses the collaborative optimization of scheduling for harvesting operations and harvest-aid vehicles on farms, with the goal of minimizing the total non-productive waiting time and coordination costs. The study takes into account both field and greenhouse scenarios, designing a multi-machine collaborative scheduling model focused on intelligent harvest-aid vehicles. The dynamic scheduling approach adopts a priority strategy based on the closest relative distance between the harvest-aid vehicle and the harvester, considering the static harvesting path and current vehicle status. When the harvest-aid vehicle reaches its load threshold, the system calculates the time for an idle vehicle to join the operation and the waiting time for the harvester, optimizing the collaboration between the harvester and the harvest-aid vehicle.

By comparing the CHVH algorithm with NSGA-II, MMOPSO, and JAYA, the results show that, for small farms, CHVH reduces the total non-productive time by 5.1%, 28.2%, and 4.9%, respectively, and lowers the coordination costs by 32.0%, 57.8%, and 31.5%. For large farms, it reduces the total non-productive time by 0.7%, 31.5%, and 5.3%, while reducing the coordination costs by 31.5%, 39.2%, and 27.5%.

This study converts the scheduling scheme into digital encoding, designing a JAYA algorithm for discrete multi-objective problems. A search strategy combining an opposition-based learning mechanism with a long-term memory library balances global and local searches. MMOPSO shares elite information through an external library, while JAYA directly uses the global best and worst solutions. JAYA's straightforward approach is theoretically more efficient, especially in quickly finding the global optimum. Although all four algorithms are multi-objective optimization algorithms, the non-dominated sorting and crowding calculation of NSGA-II are effective, but they also introduce additional computational complexity and sensitivity to parameters. There is no better performance in solving multi-machine scheduling problems under complex and multi-constraint conditions.

For farms of different scales, the average running time of our algorithm is 10.2 s, which is the shortest among the four algorithms. The MMOPSO algorithm combines particle swarm optimization and an evolutionary strategy, and its particle updates depend on the relative position and velocity between particles, which may result in a lower search efficiency than the JAYA algorithm for complex problems. NSGA-II retains superior individuals through non-dominated sorting and an elitist strategy. In contrast, the JAYA algorithm generates new solutions by directly utilizing the current global best and worst solutions. It leverages the two most promising and representative extreme points in the search space, thus providing a higher efficiency.

This paper aims to shorten total non-production waiting time and coordination costs, and designs a multi-machine collaborative scheduling algorithm. This provides a reference plan for the collaboration of harvesting and transportation. The experiment shows that the research algorithm can schedule auxiliary transport vehicles on farms of different scales. However, this study does not take into account obstacles, auxiliary transport vehicle energy, and path planning on farms. This requires improving the intelligence level of auxiliary transport vehicles and increasing the algorithm constraints, which is also one of our future research directions.

6. Conclusions

In vegetable production, various challenges exist, including low intelligence in harvesting and transportation, labor inefficiency for farmers, and non-productive waiting times for harvesters. To address these issues and coordinate vegetable harvesting and transportation, this paper aimed to minimize the non-productive waiting time and scheduling costs for harvesting assistance vehicles. Drawing on reverse learning and memory theory, a discrete multi-objective solution algorithm based on Jaya optimization was developed to optimize transportation vehicle scheduling.

By conducting experiments on a farm, the practicality and effectiveness of the proposed model and algorithm were verified. The studied algorithm enables the generation of a conflict-free harvest-aid vehicle scheduling scheme, effectively assisting harvesters and greenhouse vegetable harvesting. Compared to other algorithms, this approach yields superior solutions and exhibits a faster convergence.

Considering the complexity of the harvester operation mode and the possibility of unforeseen factors such as failures in actual vegetable production, future research will explore the modeling of emergency windows to optimize the entire vegetable production process and further enhance agricultural machinery scheduling schemes.

Author Contributions: X.H.: conceptualization, methodology, writing—original draft, writing—review and editing. H.W.: investigation, methodology, writing—original draft. H.Z.: conceptualization, validation, writing—review and editing. J.G.: software, validation. W.G.: visualization, writing—review and editing. Y.M.: validation. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by National Key R&D Program of China (grant number 2023YFD2001205); China Agriculture Research System of MOF and MARA (grant number CARS-23-D07); Reform and Development Project of Beijing Academy of Agricultural and Forestry Sciences- Research on Identification and Precise Positioning Technology for the Entire Growth Cycle of Open Field Cabbage.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The data that support the findings of this study are available from the corresponding author upon request.

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

References

1. Tang, Y.; Dong, J.; Gruda, N.; Jiang, H. China requires a sustainable transition of vegetable supply from area-dependent to yield-dependent and decreased vegetable loss and waste. *Int. J. Environ. Res. Public Health* **2023**, *20*, 1223. [[CrossRef](#)] [[PubMed](#)]
2. Yang, Q.; Ahmad, I.; Faheem, M.; Siddique, B.; Xu, H.; Addy, M. Development and assessment of belt drive seedlings transmission device for fully-automatic vegetable transplanter. *Comput. Electron. Agric.* **2021**, *182*, 105958. [[CrossRef](#)]

3. Wang, Z.H.; Xun, Y.; Wang, Y.K.; Yang, Q.H. Review of smart robots for fruit and vegetable picking in agriculture. *Int. J. Agric. Biol. Eng.* **2022**, *15*, 3–54.
4. Liu, Y.; Xia, H.; Feng, J.; Jiang, L.; Li, L.; Dong, Z.; Zhao, K.; Zhang, J. An enveloping, centering, and grabbing mechanism for harvesting hydroponic leafy vegetables cultivated in pipeline. *Agronomy* **2023**, *13*, 476. [\[CrossRef\]](#)
5. Loukatos, D.; Petrongonas, E.; Manes, K.; Kyrtopoulos, I.V.; Dimou, V.; Arvanitis, K.G. A synergy of innovative technologies towards implementing an autonomous DIY electric vehicle for harvester-assisting purposes. *Machines* **2021**, *9*, 82. [\[CrossRef\]](#)
6. Yang, Q.; Lian, Y.; Xie, W. Hierarchical planning for multiple AGVs in warehouse based on global vision. *Simul. Model. Pract. Theory* **2020**, *104*, 102124. [\[CrossRef\]](#)
7. Juan-Vásquez, P.; Fernando, A.; Autcheein, A. Workload and production assessment in the avocado harvesting process using human-robot collaborative strategies. *Biosyst. Eng.* **2022**, *223*, 56–77. [\[CrossRef\]](#)
8. Li, S.; Zhang, M.; Wang, N.; Cao, R.; Zhang, Z.; Ji, Y.; Li, H.; Wang, H. Intelligent scheduling method for multi-machine cooperative operation based on NSGA-III and improved ant colony algorithm. *Comput. Electron. Agric.* **2023**, *204*, 107532. [\[CrossRef\]](#)
9. Liu, H.; Zhang, L.; Zhao, B.; Tang, J.; Wang, F.; Wang, S. Research on Agricultural Machine Scheduling in Hilly Areas Based on Improved Non-Dominated Sorting Genetic Algorithm-III. *IEEE Access* **2024**, *12*, 32584–32596. [\[CrossRef\]](#)
10. Kan, X.; Thayer, T.C.; Carpin, S.; Karydis, K. Task planning on stochastic aisle graphs for precision agriculture. *IEEE Robot. Autom. Let.* **2021**, *6*, 3287–3294. [\[CrossRef\]](#)
11. Thayer, T.C.; Vougioukas, S.; Goldberg, K.; Carpin, S. Routing algorithms for robot-assisted precision irrigation. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018; pp. 2221–2228.
12. Peng, C.; Vougioukas, S.G. Deterministic predictive dynamic scheduling for crop-transport robots acting as harvesting aids. *Comput. Electron. Agric.* **2020**, *178*, 105702. [\[CrossRef\]](#)
13. Yang, H.; Xiong, S.; Frimpong, S.A.; Zhang, M. A Consortium Blockchain-Based Agricultural Machinery Scheduling System. *Sensors* **2020**, *20*, 2643. [\[CrossRef\]](#) [\[PubMed\]](#)
14. Pan, W.; Wang, J.; Yang, W. A Cooperative Scheduling Based on Deep Reinforcement Learning for Multi-Agricultural Machines in Emergencies. *Agriculture* **2024**, *14*, 772. [\[CrossRef\]](#)
15. Hu, Y.; Liu, Y.; Wang, Z.; Wen, J.; Li, J.; Lu, J. A two-stage dynamic capacity planning approach for agricultural machinery maintenance service with demand uncertainty. *Biosyst. Eng.* **2020**, *190*, 201–217. [\[CrossRef\]](#)
16. Fu, X.; Zhang, G.; Yuan, H.; Wang, W.; Wang, J.; Huang, Z. Swiss Round Selection Algorithm for Multi-Robot Task Scheduling. *Appl. Sci.* **2024**, *14*, 5029. [\[CrossRef\]](#)
17. Zhang, H.; Wu, Y.; Hu, J.; Wang, Y. Collaborative optimization of task scheduling and multi-agent path planning in automated warehouses. *Complex Intell. Syst.* **2023**, *9*, 5937–5948.
18. Ma, J.; Yang, S.; Jing, H. Intelligent Warehouse Robot Scheduling System Using a Modified Nondominated Sorting Algorithm. *Discrete Dyn. Nat. Soc.* **2022**, *2022*, 2021535. [\[CrossRef\]](#)
19. Hou, M.; Liu, X.; Wang, R.; Dong, C.; Sun, Q. Enhanced-Interval Optimal Scheduling of Power-Transportation Interconnected System Considering Pile (Station) Equilibrium Price. *IEEE Syst. J.* **2024**, *18*, 1320–1331. [\[CrossRef\]](#)
20. Barrientos, A.G.; Lopez, J.L.; Espinoza, E.S.; Hoyo, J.; Valencia, G. Object Transportation Using a Cooperative Mobile Multi-Robot System. *IEEE Lat. Am. Trans.* **2016**, *14*, 1184–1191. [\[CrossRef\]](#)
21. Petrović, M.; Jokić, A.; Miljković, Z.; Kulesza, Z. Multi-objective scheduling of a single mobile robot based on the grey wolf optimization algorithm. *Appl. Soft Comput.* **2022**, *131*, 109784. [\[CrossRef\]](#)
22. De Sousa SK, A.; Freire RC, S.; Carvalho EA, N.; Molina, L.; Santos, P.C.; Freire, E.O. Two-Layer Workspace: A New Approach to Cooperative Object Transportation with Obstacle Avoidance for Multi-Robot System. *IEEE Access* **2022**, *10*, 6929–6939. [\[CrossRef\]](#)
23. Ma, X.; Zhou, X. Research on the Scheduling of Mobile Robots in Mixed-Model Assembly Lines Considering Workstation Satisfaction and Energy Consumption. *IEEE Access* **2022**, *10*, 84738–84753. [\[CrossRef\]](#)
24. Liu, Z.; Wei, H.; Wang, H.; Li, H.; Wang, H. Integrated Task Allocation and Path Coordination for Large-Scale Robot Networks with Uncertainties. *IEEE T. Autom. Sci. Eng.* **2022**, *10*, 2750–2761. [\[CrossRef\]](#)
25. Ham, A. Transfer-robot task scheduling in flexible job shop. *J. Intell. Manuf.* **2020**, *31*, 1783–1793. [\[CrossRef\]](#)
26. Gultekin, H.; Gürel, S.; Taspinar, R. Bicriteria scheduling of a material handling robot in an m-machine cell to minimize the energy consumption of the robot and the cycle time. *Robot. Comput. Integr. Manuf.* **2021**, *72*, 102207. [\[CrossRef\]](#)
27. Chen, J.L.; Cao, J.N.; Liang, Z.X.; Cheng, Z.Q.; Wang, J. GraphWare: A graph-based middleware enabling multi-robot cooperation. *Concurr. Comp.-Pract. E* **2022**, *34*, e6995. [\[CrossRef\]](#)
28. Mourad, A.; Puchinger, J.; Woensel, T.V. Integrating autonomous delivery service into a passenger transportation system. *Concurr. Int. J. Prod. Res.* **2021**, *59*, 2116–2139. [\[CrossRef\]](#)
29. Rao, V.R. Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *Int. J. Ind. Eng. Comp.* **2016**, *7*, 19–34.
30. Bas, E. Solving continuous optimization problems using the improved Jaya algorithm (IJaya). *Artif. Intell. Rev.* **2022**, *55*, 2575–2639. [\[CrossRef\]](#)
31. Tefek, M.F.; Beskirli, M. JayaL: A Novel Jaya Algorithm Based on Elite Local Search for Optimization Problems. *Arab. J. Sci. Eng.* **2021**, *46*, 8925–8952. [\[CrossRef\]](#)

32. Mahdavi, S.; Rahnamayan, S.; Deb, K. Opposition-based learning: A literature review. *Swarm Evol. Comput.* **2018**, *39*, 1–23. [[CrossRef](#)]
33. Zhao, F.; Zhang, H.; Wang, L.; Ma, R.; Xu, T.; Zhu, N.; Jonrinaldi. Asurrogate-assisted Jaya algorithm based on optimal directional guidance and historical learning mechanism. *Eng. Appl. Artif. Intel.* **2022**, *111*, 104775. [[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.