scientific reports

Check for updates

OPEN

An improved artificial bee colony algorithm with path relinking strategy for the integrated distributed no-wait flow-shop scheduling and vehicle routing problem

Chenxin Dong[1,2,3] & Hui Zhang[1,2]✉

The coordinated optimization of production and transportation ensures that enterprises can lower the total cost of the supply chain while providing high-quality products to consumers, thereby increasing company profits. The integrated distributed no-wait flow-shop scheduling and vehicle routing problem (DNFSVRP) is considered for a make-to-order (MTO) manufacturer. A mathematical model with the objective of minimizing the maximum delivery time is established. For the DNFSVRP, three vectors are designed to represent the solution encoding and an improved artificial bee colony (IABC) algorithm with problem-specific knowledge is proposed. Firstly, an efficient initialization method is developed to generate high-quality initial solutions. Secondly, in the employed bee phase, eight neighborhood structures are designed to expand the search space. Moreover, in the onlooker bee phase, a path relinking strategy is adopted to fully utilize the information of the best solution and accelerate the convergence speed. Furthermore, in the scout bee phase, a destruction and reconstruction approach based on problem-specific knowledge is designed to effectively balance the breadth and depth of the search. Finally, five sets of comparative experiments are conducted, and the results demonstrate the effectiveness of the IABC algorithm in addressing the DNFSVRP.

Keywords Make-to-order manufacturer, Vehicle routing problem, Problem-specific destruction and reconstruction strategy, Path-relinking strategy

In the supply chain, production and distribution, as two crucial components, are independently managed by different departments within the company. From the perspective of the production department, increasing inventory levels leads to unnecessary inventory costs and rising production costs; from the perspective of the distribution department, effective distribution according to customer demand is a top priority. This independent management approach results in the production and distribution departments pursuing their own interest's maximization, neglecting the overall interests of the supply chain. Therefore, coordinated optimization of production and transportation ensures that enterprises can provide high-quality products to consumers while reducing the total cost of the supply chain, enhancing enterprise profitability.

Considering the integration optimization for production and distribution, Hou et al.[1] designed a multi-objective brain storm optimization algorithm to maximum production quality and minimum earliness and tardiness. Bahmani et al.[2] presented an efficient wolf optimization algorithm to address the problems of parts ordering, two-stage assembly flow shop scheduling and distribution of products through routing. Hou et al.[3] proposed an enhanced brain storm optimization algorithm to handle an integrated distributed production and distribution problem, where a mixed integer programming model with minimizing total weighted earliness and

[1]School of Intelligent Vehicles and Low-Altitude Emergency Response, Qingdao Hengxing University of Science and Technology, Qingdao, China. [2]Department of Mathematics and Yunnan Key Laboratory of Modern Analytical Mathematics and Applications, Yunnan Normal University, Kunming, China. [3]Faculty of Engineering, Built Environment, and Information Technology (FOEBEIT), SEGi University, Petaling Jaya, Malaysia. ✉email: 2323080048@ynnu.edu.cn

tardiness was established. He et al.[4] investigated an enhanced branch-and-price algorithm for the integrated production and transportation scheduling problem. Zou et al.[5] proposed an improved genetic algorithm to address the integrated production scheduling and vehicle routing with capacity constraints. Maliheh et al.[6] integrated the production and distribution problem with due dates, assignment to multiple heterogeneous vehicles and delivery of customer orders in time-windows.

Additionally, make-to-order (MTO) manufacturing refers to the practice where companies produce goods based on actual customer orders to reduce waste and enhance customer satisfaction. Yin et al.[7] addressed the scheduling problem of production and batch delivery in MTO manufacturing, considering the situation where the completion time of each job in a batch is consistent with the batch completion time. Bachtenkirch[8] investigated a novel single-stage scheduling problem, where jobs can be flexibly delivered in batches within the delivery date. Guo et al.[9] studied the integrated scheduling of production and transportation in MTO supply chains, and established a harmony search-based memory optimization model to address this problem.

Moreover, the artificial bee colony (ABC) algorithm is an optimization method that simulates the behavior of bees and is used to solve multi-variable function optimization problems. Zuo et al.[10] investigated an assembly mixed-model flow shop scheduling problem considering energy consumption, proposing an ABC algorithm based on population diversity. Zhang et al.[11] applied the ABC algorithm to solve distributed permutation flow shop scheduling problems considering sequence-dependent setup times. Zheng et al.[12] studied the multi-objective green collaborative scheduling problem. Li et al.[13] proposed a multi-objective discrete ABC algorithm to solve distributed heterogeneous flow shop scheduling problems. Li et al.[14] used the ABC algorithm to solve parallel processing distributed flow shop scheduling problems and designed a novel scout bee heuristic algorithm to enhance search performance.

From the literature analysis, it is evident that the integrated distributed production and distribution problems have not receiving sufficient attention, despite their significant relevance in real-world industry applications. To fill this gap, this research introduces an integrated problem concerning MTO distributed production and distribution to comprehensively optimize both stages. Specifically, the production stage involves scheduling operations within a distributed no-wait flow shop, while the distribution stage addresses the resolution of a multi-depot vehicle routing problem. The main contributions of this paper are as follows:

(1) Considering the problem feature, a three-dimensional vector is used to represent each solution, i.e., factory allocation, order processing sequence, and vehicle transportation route.
(2) During the employed bee phase, six types of local search based on the production stage and two based on the transportation stage are designed to explore the depth of solution space for an optimization problem.
(3) To simplify the selection process for scout bees, an improved path relinking strategy is adopted, where the swapping or insertion operations are performed based on whether the orders are at critical factories.
(4) In the scout bee phase, a destruction and reconstruction strategy is embedded to enhance the searching abilities.

This paper adopts the following structural organization. Section “Problem formulation” presents a mathematical model for the DNFSVRP. Section "The proposed algorithm" offers an intricate examination of the components incorporated into the proposed algorithm. Section "Experimental comparisons and analysis" validates the efficacy of the proposed algorithm in addressing the DNFSVRP through extensive comparative experiments. Lastly, Sect. “Experimental analysis” concludes with remarks, emphasizing three specific domains deserving further exploration and investigation.

Problem formulation

Problem description

This study investigates integrated production and distribution scheduling problem in a supply chain. The production stage has a set of $F = \{n+1, n+2, ..., n+f\}$ factories, where n is the number of customers and f is the number of factories. Each factory can be regarded as an independent no-wait flow shop in which there are m parallel identical machines, where $M = \{1,2, ..., m\}$. The set $C = \{0,1, 2, ..., n\}$ denotes customers who locate at different places, where 0 represents a dummy customer. Each customer places one order, which needs to be assigned among factories and each order is allowed to be assigned to only one factory. Orders not only have the same processing sequence on machines, but also there is no time interval between machines to meet the no-wait condition. The processing time of each order on the machine in the factory is known in advance, however, its production sequence is unknown beforehand; thereby, the completion time of the order in the production stage is unknown as well.

The distribution stage can be seen as a multi-depot vehicle routing problem aiming at delivering the orders to customers by employing vehicles with a given maximum capacity. Limited but adequate number of homogeneous vehicles $K = \{1,2, ..., n\}$ with capacity Q are available to deliver finished orders to the customers. Due to the resource consumption caused by vehicles, one objective is to minimize the number of vehicles when shipping orders to customers. The departure time of each vehicle is determined by the maximum completion time of the orders assigned to it. Figure 1 provides scheduling diagram of the problem considered.

Problem assumptions

A mathematical model is formulated to jointly consider production scheduling and vehicle routing with the aim to minimize the maximum order delivery time. To simplify the issue at hand, the following seven assumptions are adopted.

(1) Preemption is not allowed for production and delivery operations.

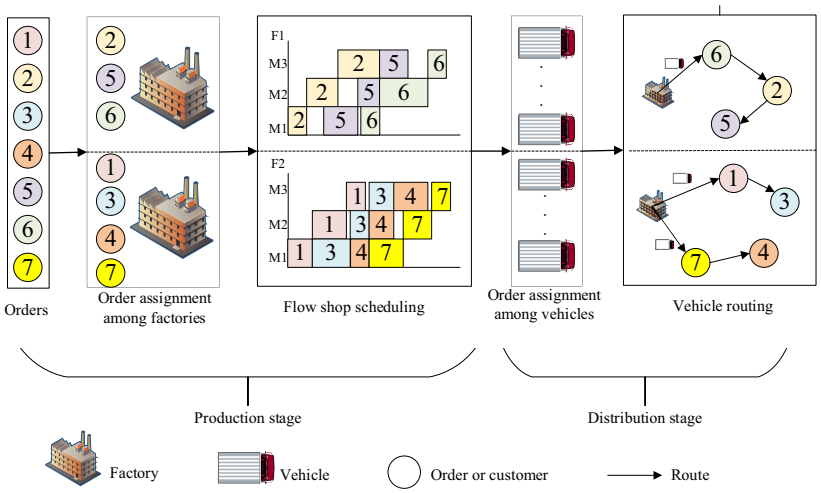

Fig. 1. Scheduling diagram of DNFSTP.

(2) All factories have the same processing capabilities, and each order can be assigned to any factory.
(3) Each order must continuously complete the entire production process in the selected factory.
(4) All parameters and required data are deterministic and available when scheduling is undertaken.
(5) The returning time of the vehicle is ignored. Since the customers only care for the delivery time of their order.
(6) Vehicles are homogeneous and sufficient vehicles are available at each factory.
(7) Each vehicle has limited capacity and the order sizes are smaller than the capacity of the vehicles.

The production processes require a detailed scheduling which means the exact allocation of the orders to the factories and determination of the sequence of the orders in the factories. The efficient transport processes also require a detailed scheduling that a schedule allocates orders to available transport vehicles and determines transport routes.

Mathematical models

Indices, parameters, and decision variables

Table 1 lists the descriptions of the indices, parameters, and decision variables used for the mathematical model of the problem considered.

Objectives

The objective function is given in Formula (1), which is used to minimize the maximum leave time of all customers or orders.

Minimize:

$$\max(L_j), \forall j \in C, j \neq 0 \tag{1}$$

Constraints related to the processing sequence

The processing sequences among orders are guaranteed as follows. Equation (2) indicates that each order has and only has one predecessor order. Constraint (3) makes sure that each order has no more than one successor processing in the same factory. Constraint (4) guarantees that the overlap between any two orders is not permitted.

$$\sum_{g \in F} \sum_{k \in C} X_{kjg} = 1, \forall j \in C, j \neq 0, k \neq j \tag{2}$$

$$\sum_{g \in F} \sum_{j \in C/\{0\}} X_{kjg} \leqslant 1, \forall k \in C, k \neq 0, k \neq j \tag{3}$$

$$\sum_{k \in C/\{0\}} X_{kjg} + X_{jkg} \leqslant 1, \forall j \in C, j \neq 0, k \neq j, \forall g \in F \tag{4}$$

Constraints related to the factory assignment

The factory assignment constraint is guaranteed as follows. Constraint (5) makes sure that each factory must be assigned at least one order. Equation (6) makes sure that any order must be assigned to a unique factory.

Notations	Descriptions
Indices	
N	Set of orders and factories, $N = C/\{0\} \cup F$.
k, j, l	Index for orders or customers.
g	Index for factories.
i	Index for machines.
h	Index for vehicles.
Parameters	
p_{ij}	Processing time of order j on machine i.
q_j	Size of order j.
t_{kj}	Transportation time between customers k and j.
T_{gj}	Transportation time between the manufacturing factory g and customer j.
v_j	Service time at customer j.
G	An infinite constant.
Decision variables	
X_{kjg}	Binary variable. It equals to 1 if order j is processed consecutively after order k at factory g, and 0 otherwise.
Y_{kjgh}	Binary variable. It equals to 1 if vehicle h goes from customer k to j at factory g, and 0 otherwise.
Z_{jg}	Binary variable. It equals to 1 if order j in factory g, and 0 otherwise.
S_{jig}	Start time of order j on machine i at factory g.
C_{jig}	Completion time of order j on machine i at factory g.
C_g	Completion time of factory g.
V_{gh}	Departure time of vehicle h at factory g.
U_{ghj}	Delivery time of order j on vehicle h at factory g.
A_j	Arrival time at customer j.
L_j	Leave time from customer j.

Table 1. Indicates, parameters and decision variables.

$$\sum_{j \in C \setminus \{0\}} Z_{jg} \geqslant 1, \forall g \in F \tag{5}$$

$$\sum_{g \in F} Z_{jg} = 1, \forall j \in C, j \neq 0 \tag{6}$$

Constraints related to the start and completion times

The start and completion times are calculated as follows. Constraint (7) is used to calculate the completion time of the first operation of order j, where p_{1j} is the processing time of the first operation of order j. Constraint (8) makes sure that the overlap between two consecutive orders on the same machine is not permitted, that is, the successor can be started until the completion of the predecessor order. The no-wait constraint is guaranteed by the Constraints (9)-(10), while Constraint (11) calculates the maximum completion time for each factory.

$$C_{j1g} \geqslant p_{1j} - G \cdot (1 - Z_{jg}), \forall j \in C, j \neq 0, \forall g \in F \tag{7}$$

$$C_{jig} \geqslant C_{kig} + p_{ij} - G \cdot (1 - X_{kjg}), \forall k, j \in C, k, j \neq 0, k \neq j, \forall i \in M, \forall g \in F \tag{8}$$

$$S_{jig} = C_{j(i-1)g}, \forall j \in C, j \neq 0, \forall i \in M, i \neq 1, \forall g \in F \tag{9}$$

$$C_{jig} = S_{jig} + p_{ij}, \forall j \in C, j \neq 0, \forall i \in M, \forall g \in F \tag{10}$$

$$C_g \geqslant C_{jmg} - G \cdot (1 - Z_{jg}), \forall j \in C, j \neq 0, \forall g \in F \tag{11}$$

Constraints related to the delivery and transportation

Constraint (12) makes sure the continuity between the production and delivery stages. Equation (13) guarantees that every customer must be visited once. Constraint (14) specifies that a vehicle departs its associated factory at most once. Constraint (15) implies that the departure time of orders in the delivery stage is greater than or equal to their completion time in the production stage. Constraint (16) guarantees that the departure time of vehicles must be greater than or equal to delivery time of orders in vehicles. Constraints (17)–(18) define the arrival time of vehicles at customers. Constraint (19) ensures that the leave time of vehicles from customers.

$$\sum_{k\in C} X_{kjg} = \sum_{h\in K}\sum_{l\in C} Y_{ljgh}, \forall j \in C, j \neq 0, \forall g \in F \tag{12}$$

$$\sum_{g\in F}\sum_{h\in K}\sum_{k\in N} Y_{kjgh} = 1, \forall j \in C, j \neq 0 \tag{13}$$

$$\sum_{j\in C\setminus\{0\}} Y_{kjgh} \leqslant 1, \forall k \in F, \forall g \in F, \forall h \in K \tag{14}$$

$$U_{ghj} \geqslant C_{jmg} - G \cdot (1 - \sum_{k\in C} Y_{kjgh}), \forall j \in C, j \neq 0, \forall g \in F, \forall h \in K \tag{15}$$

$$V_{gh} \geqslant U_{ghj}, \forall j \in C, j \neq 0, \forall g \in F, \forall h \in K \tag{16}$$

$$A_j \geqslant V_{gh} + T_{gj} - G \cdot (1 - \sum_{k\in F} Y_{kjgh}), \forall j \in C, j \neq 0, \forall g \in F, \forall h \in K \tag{17}$$

$$A_j \geqslant L_k + t_{kj} - G \cdot (1 - \sum_{g\in F}\sum_{h\in K} Y_{kjgh}), \forall k, j \in C, k, j \neq 0 \tag{18}$$

$$L_j \geqslant A_j + v_j, \forall j \in C, j \neq 0 \tag{19}$$

Constraints related to the variable range

Constraint (20) makes sure that the maximum capacity of the vehicle needs to be considered when loading it. Constraint (21)–(26) limit the range of decision variable.

$$\sum_{k\in N}\sum_{j\in C/\{0\}} Y_{kjgh} \cdot q_j \leqslant Q, \forall g \in F, \forall h \in K \tag{20}$$

$$S_{ijg} \geqslant 0, \forall i \in M, \forall j \in C, \forall g \in F \tag{21}$$

$$C_{ijg} \geqslant 0, \forall i \in M, \forall j \in C, \forall g \in F \tag{22}$$

$$U_{ghj} \geqslant 0, \forall g \in F, \forall h \in K, \forall j \in C, j \neq 0 \tag{23}$$

$$A_j \geqslant 0, \forall j \in C \tag{24}$$

$$L_j \geqslant 0, \forall j \in C \tag{25}$$

$$V_{gh} \geqslant 0, \forall g \in F, \forall h \in K \tag{26}$$

The proposed algorithm

This section presents the IABC algorithm used to solve the DNFSVRP considered. The main task of the employed bees is to perform local searches on each solution. A variable neighborhood descent (VND) algorithm with eight types of neighborhood structures is embedded to enhance local search capabilities. The scout bees perform the search task around the solutions found by the employed bees. To simplify the selection method for scout bees, a path relinking strategy is adopted. Through swapping and insertion operations on the current solution, it gradually approaches the better solution found by the employed bees. The scout bees are used to replace solutions that have reached a specified number of iterations without improvement. Compared to the classical ABC algorithm, which randomly generates a solution for the scout bees, the IABC algorithm utilizes a set of problem-specific destruction and reconstruction operations. Algorithm 1 shows the framework of the IABC.

Set parameters the number of food source (*SN*) and trials (*limit*)
Initialize the population using the ECT-2-machine method proposed in Section 3.2
Employed bee phase
    For *i*=1,2,...,*PS*
        k=rand(1,8)
        Produce a new solution π_i^* for the *i*th employed bee who is associated with solution π_i by using the *k*th neighborhood structure proposed in Section 3.3 and evaluate the new solution
        If π_i^* is better than or equal to π_i, let $\pi_i = \pi_i^*$
    End **For**
Onlooker phase
    For *i*=1,2,...,*PS*
        Select a food source for the onlooker bee *i* by using the path relinking strategy presented in Section 3.4 and evaluate
        If the generated solution is better than or equal to the selected one, update the population.
    End **For**
Scout phase
    If a solution in the population has not been improved during the last *limit* number of trials, abandon it and put a new solution generated by performing destruction and reconstruction strategy presented in 3.5.

Algorithm 1. The framework of IABC. Input: a population. Output: the best agent.

Representation of solution

In the DNFSVRP, a complete solution contains the following information: orders assignment among factories, orders processing sequence in each factory at the production stage, as well as customers delivery routes at the distribution stage. The length of the first vector is equal to the total number of orders. Each element of the first vector is represented by a factory number. The second vector is represented by a two-dimensional array, where the length of the first dimension is the number of factories, and the second dimension is the processing sequence of the orders in each factory. The vehicles departing from factories are loaded with some orders for

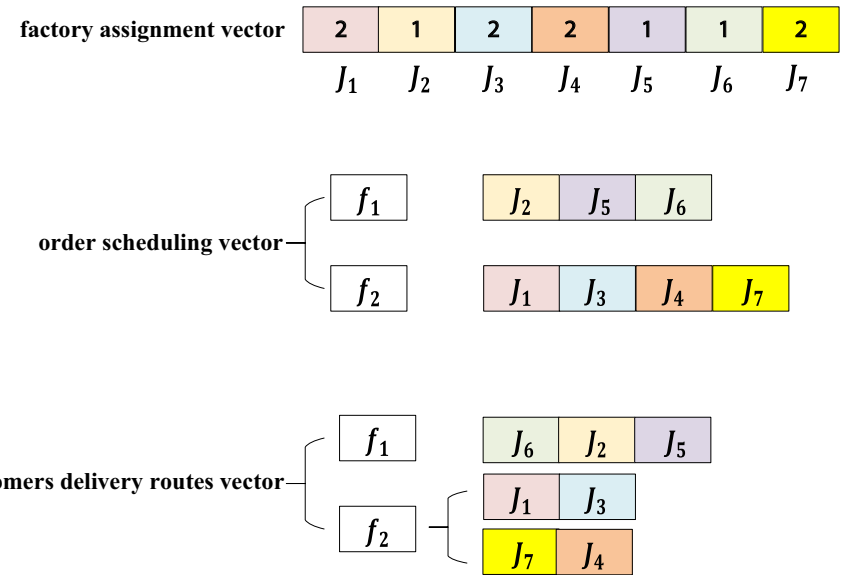

Fig. 2. Solution representation in DNFSVRP.

transportation to customers, and the third vector represents delivery routes of the vehicles between customers. Figure 2 shows a solution representation example.

Initialization phase

To solve the integrated production and distribution problem, an initial population with a high level of quality and diversity always gains a satisfying result in the competitive levels of performance. Hence, to construct an initial population, an earliest completion time (ECT)−2-machine initialization strategy is embedded in the IABC algorithm. Firstly, all orders are sorted as non-decreasing order according to total processing time. Subsequently, the orders are allocated to the factory with the earliest completion time after inserting the order. Finally, the S. M. Johnson n-job, 2-machine algorithm[15] is used for the processing sequence of each factory. Equation (27) accumulates the processing time on the first k machines for the jth order in the kth loop, and Eq. (28) accumulates the processing time on the last k machines for the jth order in the kth loop, where k ranges from 1 to $m-1$. Thus, a total of $m-1$ feasible solutions is generated. In the kth loop, based on the calculation of each job, a total of $2n$ numbers are obtained. The minimum number is selected from these $2n$ numbers. If the minimum number is obtained from Eq. (27), the corresponding job is placed at the front of the processing sequence; otherwise, it is placed at the end of the processing sequence. Remove the selected orders, and repeat the above process until all orders are placed in the sequence.

$$\theta_{j1}^{k} = \sum_{i=1}^{k} p_{ij} \tag{27}$$

$$\theta_{j2}^{k} = \sum_{i=m+1-k}^{m} p_{ij} \tag{28}$$

The flow chart of n-job, 2-machine algorithm is shown in Fig. 3. And an example of a 7-order, 5-machine problem is provided in Table 2. Let $k=3$, Eq. (27) is used to calculate the sum of processing times on machines a, b, c, while Eq. (28) calculates the sum of processing times on machines c, d, e. Table 3 shows the results obtained by the two Equations. Among the results, the 6th order calculated by Eq. (28) has the smallest value, therefore, the order 6 is taken out and placed in the last position of the sequence. Next, order 1 is chosen and placed in the first position of the sequence. Continuing until all jobs have been assigned a sequence position yields the sequence {1, 3, 7, 2, 5, 4, 6}.

Local reinforcement strategy

VND algorithm has a simple structure and uses different neighborhood structures to systematically switch from one neighborhood to another. Owing to the simplicity and flexibility of its framework, it can easily integrate many neighborhood search operators, which ensures that VND can be used to solve many different types of optimization problems, such as flow shop scheduling problem, facility layout problem, traveling salesman problem, project scheduling problem, vehicle routing problems[16–19]. In the procedure of VND, it is crucial to define effective neighborhood searches. Eight types of neighborhood structures are designed to perform exploitation of solution space in different regions. The detailed descriptions of these neighborhoods are as follows:

One point swap neighborhood First, a random order in the critical factory are chosen. Then, the two subsequences before and after that order are swapped. Figure 4 represents a schematic of this operator.

Reverse neighborhood First, two orders in the critical factory are selected randomly and then the orders between them are reversed. Figure 5 represents a schematic of this operator.

Three points swap neighborhood First, three random orders in the critical factory are selected. Then, swap these three orders in a random way. Next, the best neighborhood solution is selected by measuring their fitness. Figure 5 represents a schematic of this operator.

Insert in multi factory neighborhood First, randomly select an order from the critical factory. Then, insert it into the best position of a randomly chosen non-critical factory. Figure 6 represents a schematic of insert in multi factory.

Swap-cyclic neighborhood In this neighborhood search strategy, starting from the first factory, randomly select an order from the factory and exchange it with any order from the next factory until the last factory. The order from the last factory is swapped with the order from the first factory. Figure 7 represents a schematic of this operator.

Insert-cyclic neighborhood This method is quite similar to the swap-cyclic neighborhood, starting from the first factory, randomly select an order from the factory and insert it to the best position in the next factory, all the way to the last factory. The order from the last factory is inserted into the first factory to minimize its fitness value after inserting the order. Figure 9 represents a schematic of this operator.

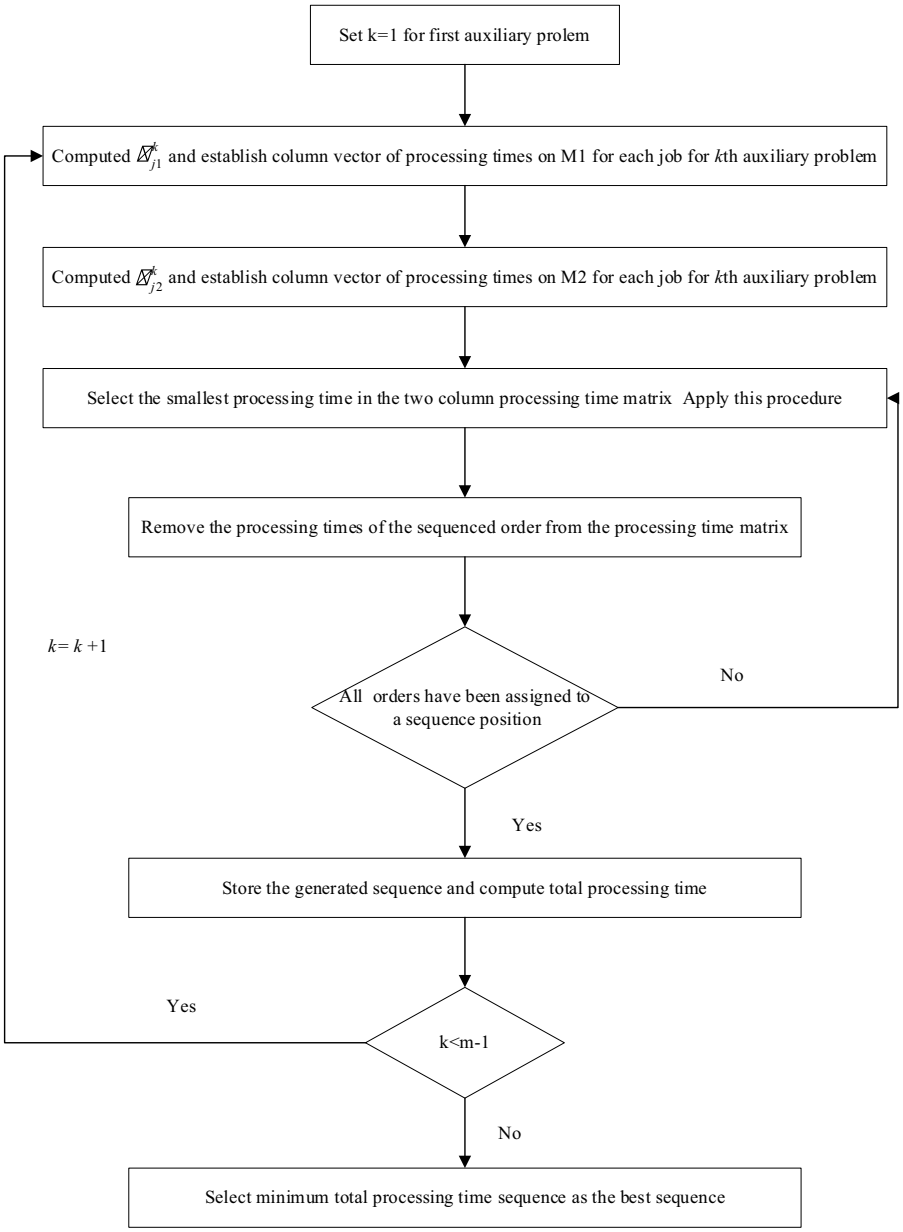

Fig. 3. The flow chart of n-job, 2-machine algorithm.

Orders	a	b	c	d	e
1	13	23	27	33	14
2	31	13	47	23	36
3	17	11	39	18	22
4	19	32	24	25	15
5	43	28	30	27	18
6	29	41	16	10	33
7	23	19	35	26	24

Table 2. Example of a 7-order, 5-machine problem.

Orders	Equation (27)	Equation (28)
1	63	74
2	91	106
3	67	79
4	75	64
5	101	75
6	86	59
7	77	85

Table 3. Processing times matrix of $k = 3$.

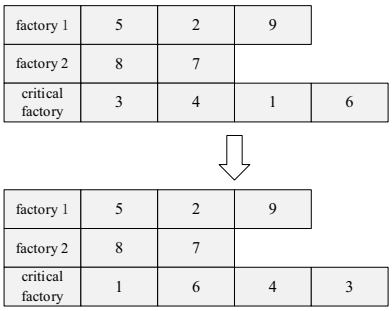

Fig. 4. Example of one point crossover.

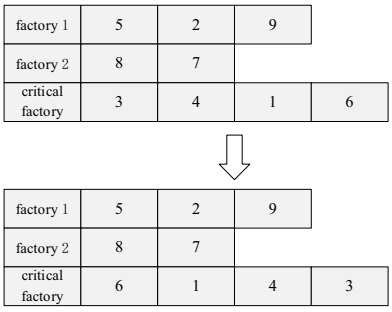

Fig. 5. Example of reverse.

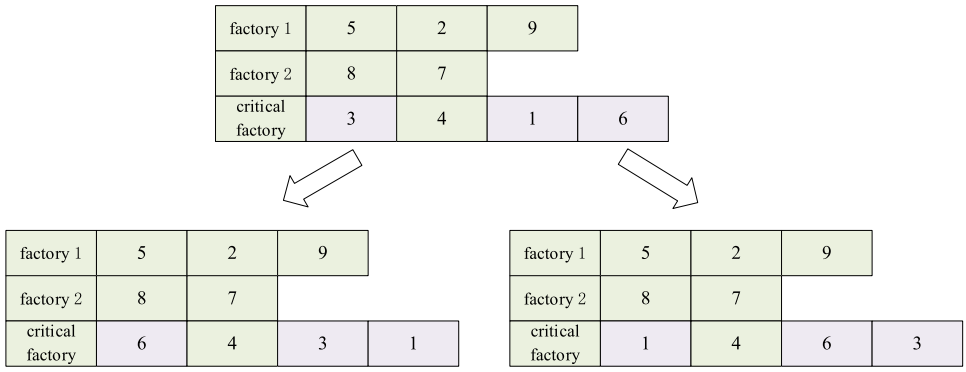

Fig. 6. Example of three points swap.

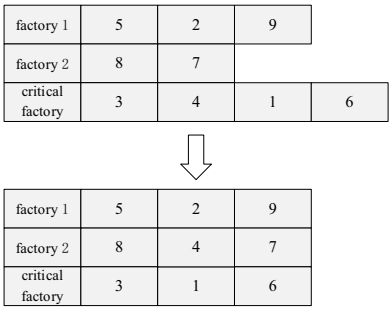

Fig. 7. Example of insert.

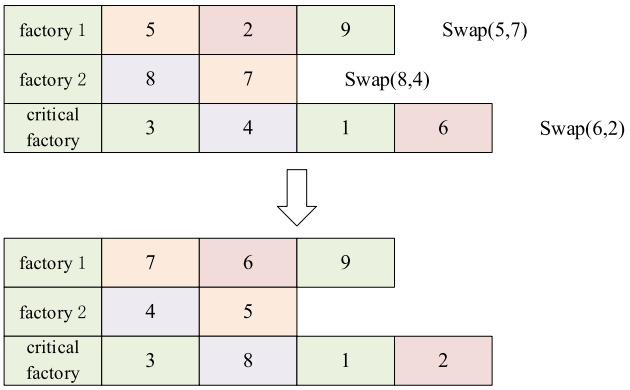

Fig. 8. Example of swap-cyclic.

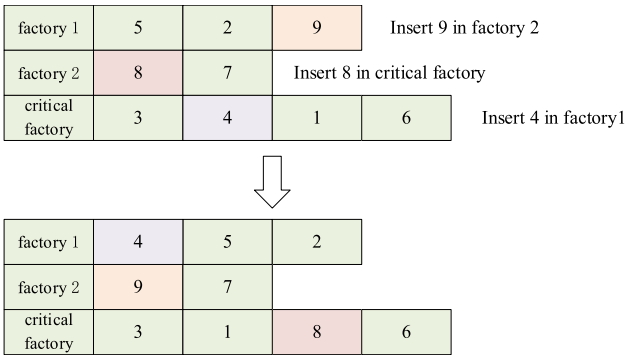

Fig. 9. Example of insert-cyclic.

Route-swap neighborhood In this policy, first, find the longest route to complete the order transportation. Then, choose a route that starts from the same factory as the longest route. Finally, take one order from each of the two paths for exchange. Figure 10 represents a schematic of this operator.

Route-insert neighborhood To conduct this approach, at first find the last route that starts from the same factory as the longest route. And then select an order from the longest route. If inserting the order into the last route does not exceed the maximum capacity of the vehicle, execute the insertion operator. Otherwise, assign a separate vehicle to the order. Figure 11 represents a schematic of route-insert.

As we know, due to the fact that vehicles departing from a critical factory have the latest departure time. Therefore, the first four neighborhood structures revolve around the critical factory for order swap, insert and reverse. To enhance the quality of solution for the optimum and maintain the diversity of the population, swap in cyclic and insert in cyclic enabled every factory, not only critical factory, to perform exchange or insertion operations. The last two neighborhood structures consider the route sequence in the transportation stage.

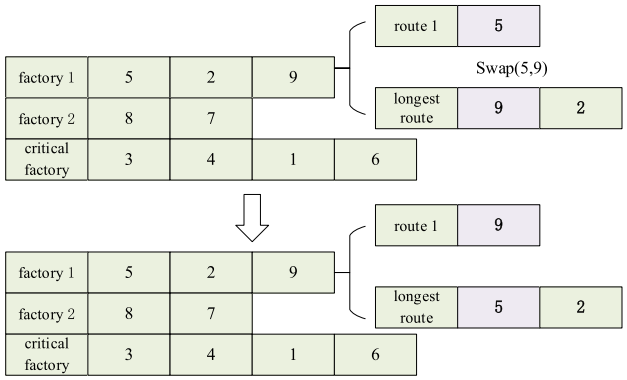

Fig. 10. Example of route-swap.

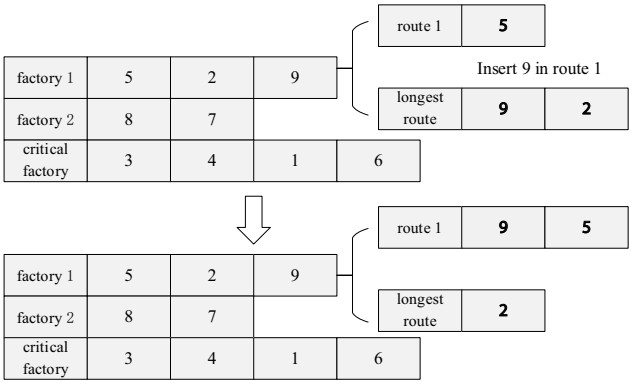

Fig. 11. Example of route-insert.

Path relinking strategy

Path relinking is a centralized search strategy used to trace the trajectory of high-quality solutions in combinatorial optimization problems. It generates new solutions by searching the neighborhoods of two given solutions. Path relinking is often combined with other heuristic algorithms or metaheuristic algorithms, playing an important role in improving both the solution speed and quality of algorithms.

The steps of path relinking are as follows: (1) select an optimal solution and compare it with the current solution; (2) select the jobs that differ between the current solution and the optimal solution; (3) if a job is in a critical factory, perform an insertion operation; otherwise, perform a swap operation; (4) repeat the above three steps until the job sequences of the current solution and the optimal solution are identical. Each execution generates an intermediate solution; and (5) calculate the fitness value of all intermediate solutions and keep the solution with the minimum fitness value.

Destruction and reconstruction operations

In the destruction stage, because the number of orders per factory is different, the unified destruction size is no longer applicable to the current problem. Number of orders per factory are presented to determine how many orders are randomly extracted from each factory. N_s is the number of orders allocated in factory *s*, and D_s is the destruction size of factory *s*. The relationship between the two is shown in the formula (29).

$$D_s = \left\lfloor \sqrt{N_s} \right\rfloor \tag{29}$$

Figure 12 provides a simple example. Factory 1 originally has three orders; it is determined that one order needs to be removed after calculation. And factory 2 needs to remove two orders. Orders removed from all factories are placed in a temporary partial sequence γ. When an order is deleted from the factory sequence, it is indispensable to remove the order from the route sequence. At the same time, the remaining loadable capacity of the vehicle also needs to be changed accordingly.

In the reconstruction stage, the orders in γ are reinserted into the factory sequence and route sequence one by one. Firstly, an order is reinserted into all the positions in all processing factories, and the position with the minimum fitness value is decided to insert the order. Afterwards, a vehicle is selected from the factory, where the order was inserted in the previous step, that can load the order without exceeding the maximum capacity of the vehicle. Finally, the order is inserted into the corresponding route sequence of the vehicle. It is worth mentioning

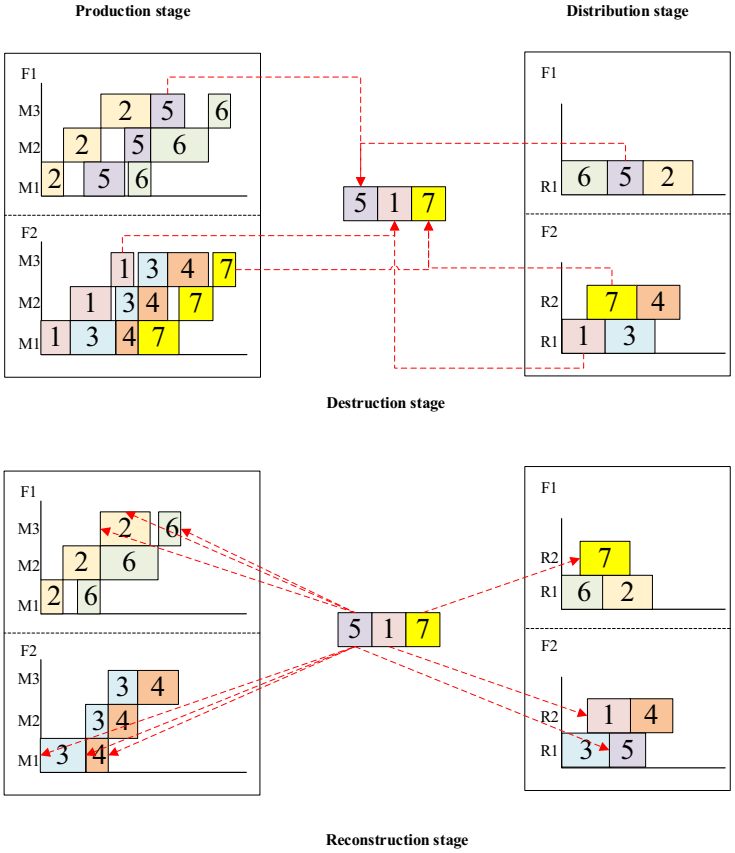

Fig. 12. Example of destruction-reconstruction.

that if there are no vehicles available to load the order, the factory is allocating an idle vehicle to transport the order to the customer.

Experimental comparisons and analysis

To evaluate the performance of the IABC algorithm, it was coded in Visual Studio 2019 and executed on a computer equipped with an Intel Core i5-7300HQ central processing unit (CPU) operating at a frequency of 2.50 GHz, 8GB of RAM, and the Windows 10 operating system.

To the best of our knowledge, there is now published study considering the integration optimization of distributed production and distribution scheduling problem with no-wait constraint and VRP transportation. Therefore, we randomly generate 30 different-sized instances to evaluate the effectiveness of the IABC, where the number of customers $n = \{20, 50, 80, 100, 200\}$, the number of factories $f = \{2, 5\}$, the number of machines per factory $m = \{2, 5, 8\}$, the processing time for each operation of workpieces in the production stage is randomly selected between [40, 50], the travel time from factories to customers and between customers is randomly selected between[10,20], the service time for each customer is randomly selected between[10,20], and considering vehicle capacity constraints, the weight of each order is randomly selected between[15,20]. Two metrics including relative percentage increase (PRI) and average relative percent difference (ARPD) are used to make detailed comparisons, which are calculated in Eqs. (30) and (31), respectively. In the two equations, c_i represents the value obtained by the ith comparison algorithm, while c_{opt} represents the best value found by all the comparison algorithms.

$$RPI = \frac{(c_i - c_{opt})}{c_{opt}} \cdot 100 \tag{30}$$

$$ARPD = \frac{1}{R}\sum_{i=1}^{R}\frac{c_i - c_{opt}}{c_{opt}} \times 100 \tag{31}$$

Effectiveness analysis of IABC components

Compared to the classical ABC algorithm, the IABC adopts the proposed initialization strategy, local search strategy, path relinking strategy, and problem-specific destruction and reconstruction strategies. To validate the performance of the IABC algorithm, experiments were conducted comparing IABC and the algorithms without the corresponding component. The results are summarized in Table 4, where the first column lists the scale of

Instances	RPI				
	IABC	I-ECT	I-VND	I-PR	I-DR
20_2_2	**0.00**	30.74	29.22	28.95	30.76
20_2_5	**0.00**	28.01	25.29	26.58	28.32
20_2_8	**0.00**	23.78	23.55	23.21	25.52
20_5_2	**0.00**	22.90	19.68	19.73	33.06
20_5_5	**0.00**	24.89	24.48	21.36	33.75
20_5_8	**0.00**	22.40	21.57	21.71	27.60
50_2_2	**0.00**	27.29	22.49	24.92	26.25
50_2_5	**0.00**	23.54	23.00	22.88	24.05
50_2_8	**0.00**	25.50	23.99	23.30	24.43
50_5_2	**0.00**	24.46	21.30	20.89	29.18
50_5_5	**0.00**	24.19	24.22	24.66	29.14
50_5_8	**0.00**	28.72	22.80	23.95	27.19
80_2_2	**0.00**	19.62	22.57	21.97	24.14
80_2_5	**0.00**	25.04	22.72	23.56	22.44
80_2_8	**0.00**	24.60	22.72	24.04	23.61
80_5_2	**0.00**	26.70	23.08	21.63	24.01
80_5_5	**0.00**	25.51	25.75	25.54	27.14
80_5_8	**0.00**	26.69	24.58	24.21	26.39
100_2_2	**0.00**	22.04	20.59	19.82	23.62
100_2_5	**0.00**	21.32	21.86	22.88	23.38
100_2_8	**0.00**	24.52	21.67	23.09	24.03
100_5_2	**0.00**	25.60	21.19	23.14	24.30
100_5_5	**0.00**	26.24	24.99	21.55	24.34
100_5_8	**0.00**	24.66	23.21	24.18	25.98
200_2_2	**0.00**	21.92	22.69	23.45	23.04
200_2_5	**0.00**	22.74	23.28	23.86	24.09
200_2_8	**0.00**	25.31	25.89	23.34	24.64
200_5_2	**0.00**	25.80	24.50	24.10	24.31
200_5_5	**0.00**	24.50	23.29	23.65	24.63
200_5_8	**0.00**	25.57	25.48	24.51	26.50
Average	**0.00**	24.83	23.39	23.36	25.99

Table 4. The computational results of four components.

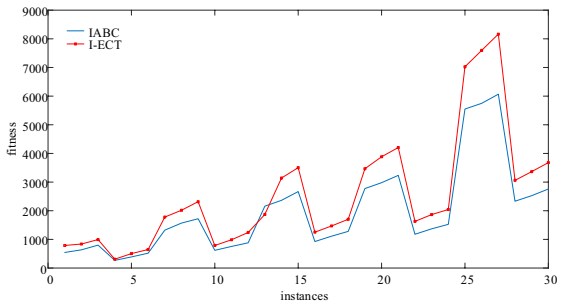

Fig. 13. The line chart of IABC, I-ECT.

different instances, namely the number of orders, the number of factories, and the number of machines in each factory. The subsequent five columns in the table represent the calculated RPI values. Detailed explanations of Table 4 can be found in Sect. "Effectiveness Analysis of Initialization Strategy"–"Effectiveness analysis of destruction and reconstruction strategy".

Effectiveness analysis of initialization strategy

To verify the effectiveness of the proposed initialization strategy, the IABC algorithm is compared with I-ECT, which adopts a random initialization strategy. Apart from the difference in initialization methods, both algorithms employ the same strategies in the employed bees, onlooker bees, and scout bees stages. Line charts

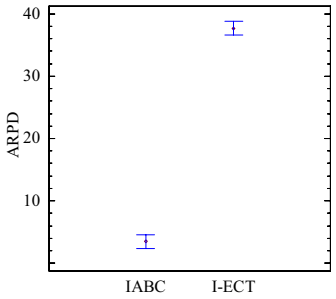

Fig. 14. ARPD analysis of IABC, I-ECT

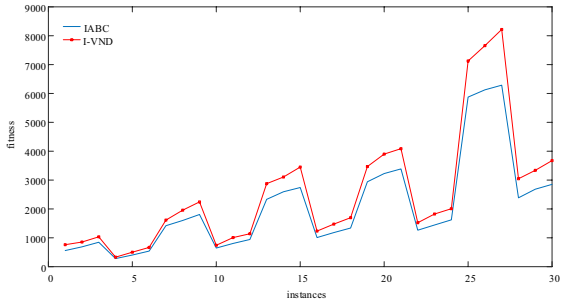

Fig. 15. The line chart of IABC, I-VND.

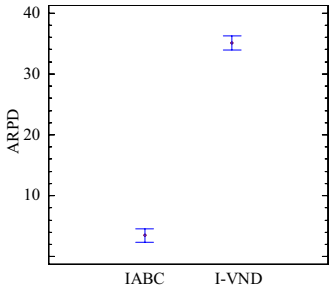

Fig. 16. ARPD analysis of IABC, I-VND.

(Fig. 13) are plotted based on the optimal solutions obtained by the algorithms, while Fig. 14 presents the Analysis of Variance (ANOVA) analysis of the ARPD results from five runs of the algorithms. The following conclusions can be drawn from the results: (1) Among the 30 instances, IABC obtains all the optimal solutions, which is significantly better than I-ECT. (2) The ANOVA comparison of the APRD values also verify the competitive performance of the IABC algorithm. This validates the effectiveness of the proposed initialization strategy.

Effectiveness analysis of local search strategy

The effectiveness of the local search strategy is analyzed by comparing IABC with the I-VND algorithm, which does not employ local search strategy in the employed bee stage. The RPI values obtained by the two algorithms are summarized in columns 2 and 4 of Table 4. To present the experimental results more clearly, Figs. 15 and 16 are plotted based on the results of running 5 times for 30 instances.

From the results, the following conclusions can be drawn: (1) The average RPI value of IABC is 0.00, whereas the average RPI value of I-VND is 23.39, significantly higher than that of IABC. (2) Observing the RPI values and line charts of the 30 instances, it is evident that all 30 solutions obtained by IABC are superior to those obtained by I-VND. (3) Fig. 16 provides an analysis of the ARPD of the experimental data, further validating that local search can effectively improve algorithm performance.

Effectiveness analysis of path relinking strategy

The effectiveness of the path relinking strategy is analyzed by comparing the IABC with I-PR, which other parameters remain the same except for the usage of the path relinking strategy. The experimental results are summarized in columns 2 and 5 of Table 4, as well as in Figs. 17 and 18.

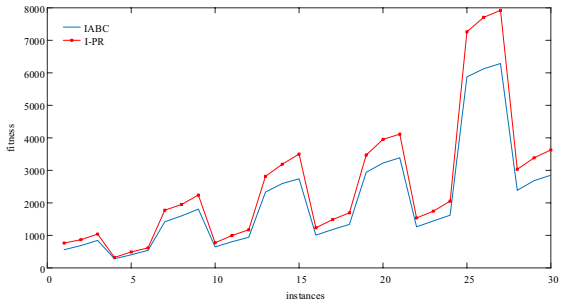

Fig. 17. The line chart of IABC, I-PR.

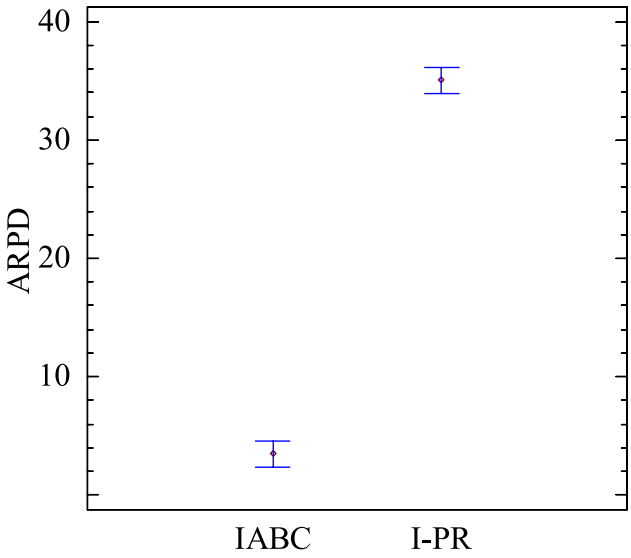

Fig. 18. ARPD analysis of IABC, I-PR.

It can be observed that: (1) The RPI values obtained by the IABC algorithm are significantly lower than those of the I-PR algorithm. (2) The last row in the table presents the average RPI values, revealing that the average RPI value of the IABC algorithm is lower than that of the I-PR algorithm. (3) In the line charts and ARPD analysis graph, the segments corresponding to the IABC algorithm are all below those of the I-PR algorithm. Therefore, for the proposed problem, the IABC algorithm outperforms the I-PR algorithm in terms of optimization performance.

Effectiveness analysis of destruction and reconstruction strategy

In the improved artificial bee colony algorithm, a problem-specific destruction and reconstruction strategy is designed in the scout bee phase to enhance global search capability. To verify the effectiveness of the destruction and reconstruction strategy, the IABC algorithm is compared with the IABC algorithm without using the destruction and reconstruction strategy (denoted as I-DR). Except for the usage of the destruction and reconstruction strategy, both IABC and I-DR algorithms use the same experimental parameters. After testing each case, the average values are collected and compared, summarized in columns 2 and 6 of Table 4. Based on the fitness values obtained from the runs, Figs. 19 and 20 are plotted.

It can be observed that: (1) In all cases, the IABC algorithm obtains 30 optimal solutions, while I-DR does not achieve optimal solutions. (2) The last row in the table, which presents the average RPI values, also indicates that the performance of IABC is superior to that of I-DR. (3) In the ARPD graph, the results obtained by the IABC algorithm are below 10, while those obtained by the I-DR algorithm are around 40, significantly higher than the results of the IABC algorithm, further confirming that the problem-specific destruction and reconstruction strategy can effectively prevent premature convergence of the algorithm.

Comparison with other effective algorithms

To validate the effectiveness of the proposed IABC algorithm in solving the DNFSVPR, it is compared with the Gray Wolf Optimization (GWO) algorithm[2], Genetic Algorithm (GA)[5], and Particle Swarm Optimization (PSO)[6]. The reasons for selecting these comparison algorithms are as follows: (1) The GWO algorithm is used to solve the distributed production and transport coordination optimization problem, but it does not

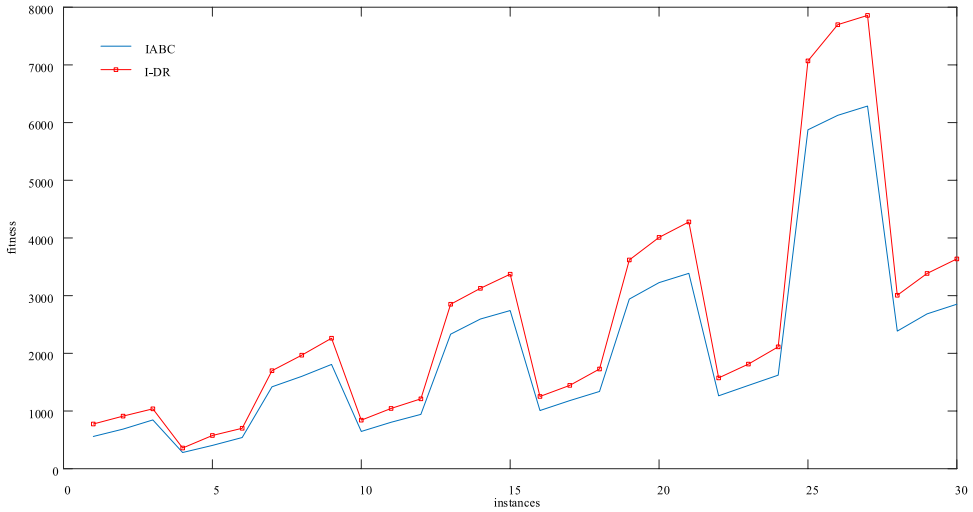

Fig. 19. The line chart of IABC, I-DR.

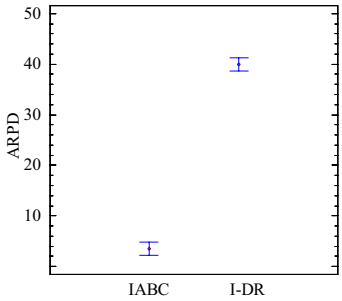

Fig. 20. ARPD analysis of IABC, I-PR.

consider no-wait constraint in the production phase. (2) The GA is employed to address MTO manufacturing and transportation problem, thus GA can be easily applied to the DNFSVRP. (3) The PSO is used to solve the integrated production and transportation problem considering vehicle routes, which is similar to the problem considered in this study. For the four compared algorithms, 30 identical test cases are used, and each case is run five times on the same computer using the same encoding mechanism and stopping criteria.

Based on the experimental results shown in Table 5; Figs. 21,22, the following conclusions can be drawn: (1) the proposed IABC algorithm obtained 21 better results out of the given 30 different-scale instances, which shows significantly performance; (2) the average performance shown in the last line in Table 5 further verify the robustness of the proposed algorithm; (3) as shown in Fig. 21, among the 30 test cases, IABC obtains the highest number of optimal solutions; (4) the average RPI values calculated for IABC, PSO, GA, and GWO are 0.010, 0.070, 0.066, and 0.040, respectively, with IABC having the lowest average RPI value; and (5) From Fig. 22, it can be observed that the IABC algorithm is located at the bottom, further demonstrating the effectiveness of the IABC algorithm in solving the DNFSVRP.

Experimental analysis

From the experimental comparisons, it can be seen that the proposed algorithm is competitive for solving the problem considered. The flow shop scheduling problems with no-wait, distributed and VRP transportation are first modeled and investigated in this study, which can be applied in many types of applications, such as prefabricated component production system, and steel and iron making systems. However, there are still many limitations which can be researched further, such as how to formulate a more robust model to combine the scheduling and transportation efficiently, how to tackle the optimization problem where one customer has more than one orders, how to design an efficient optimization algorithm to incorporate more features of scheduling and transportation, and how to apply the proposed algorithm to solve other types of scheduling problems.

Conclusion and future research

For the DNFSVRP, a mathematical model with maximum delivery time as the optimization objective is established. And the IABC algorithm with problem-specific knowledge is enhanced as follows: the ECT-2-machine method is proposed to generate high-quality initial solutions; eight neighborhood structures are designed in the scout bee phase to expand the search scope; a path relinking strategy is employed in the follower

Instances	fitness				
	Best	IABC	PSO	GA	GWO
20_2_2	544.00	**544.00**	651.00	599.00	609.00
20_2_5	636.00	**636.00**	728.00	675.00	750.00
20_2_8	800.00	**800.00**	874.00	844.00	894.00
20_5_2	266.00	**268.00**	288.00	266.00	323.00
20_5_5	388.00	**388.00**	410.00	409.00	451.00
20_5_8	519.00	**519.00**	522.00	539.00	554.00
50_2_2	1323.00	**1323.00**	1422.00	1350.00	1444.00
50_2_5	1569.00	**1569.00**	1666.00	1624.00	1592.00
50_2_8	1721.00	**1721.00**	1776.00	1835.00	1750.00
50_5_2	618.00	**618.00**	664.00	643.00	634.00
50_5_5	754.00	**754.00**	819.00	770.00	767.00
50_5_8	881.00	**881.00**	953.00	911.00	903.00
80_2_2	2160.00	**2160.00**	2240.00	2269.00	2311.00
80_2_5	2364.00	**2364.00**	2452.00	2435.00	2461.00
80_2_8	2626.00	2669.00	2726.00	2695.00	**2626.00**
80_5_2	926.00	**926.00**	1019.00	994.00	972.00
80_5_5	1112.00	**1112.00**	1136.00	1174.00	1118.00
80_5_8	1265.00	1280.00	1319.00	1293.00	**1265.00**
100_2_2	2609.00	2774.00	**2609.00**	2755.00	2866.00
100_2_5	2979.00	**2979.00**	3099.00	3098.00	3056.00
100_2_8	3179.00	3234.00	**3179.00**	3352.00	3183.00
100_5_2	1179.00	**1179.00**	1187.00	1246.00	1203.00
100_5_5	1346.00	1368.00	1429.00	1403.00	**1346.00**
100_5_8	1482.00	1525.00	1578.00	1626.00	**1482.00**
200_2_2	5431.00	5550.00	**5431.00**	5769.00	5818.00
200_2_5	5748.00	**5748.00**	6048.00	6050.00	6007.00
200_2_8	6068.00	**6068.00**	6315.00	6320.00	6141.00
200_5_2	2333.00	**2333.00**	2418.00	2398.00	2372.00
200_5_5	2509.00	2526.00	2590.00	2639.00	**2509.00**
200_5_8	2672.00	2758.00	2827.00	2909.00	**2672.00**
Average	1933.57	**1952.47**	2012.50	2029.67	2002.63

Table 5. Comparison results among IABC, GA, PSO, and GWO.

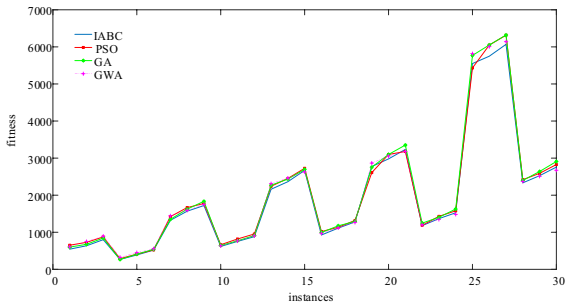

Fig. 21. The line chart of four algorithms.

bee phase to fully utilize the information of optimal solutions and accelerate the convergence speed of the algorithm; a destruction and reconstruction operation based on problem-specific knowledge is designed in the scout bee phase to effectively balance the breadth and depth of the search. Five sets of comparative experiments are designed, measured by the RPI and ARPD indicators, and the results demonstrate the effectiveness of the IABC algorithm in addressing the DNFSVRP.

Indicating a rapidly growing interest in the integrated production and distribution problem. We have provided results to some previously unresolved problems and considered extensions of some existing problems. (1) Distributed factories can consider heterogeneous factory scheduling, where variations in the processing capabilities and resources among factories are an important area for future research. (2) In this study, all

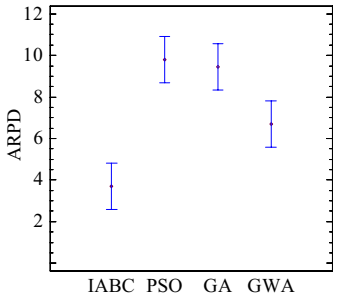

Fig. 22. ARPD analysis of four algorithms.

parameters are deterministic. However, in practical production and transportation scenarios, some parameters are uncertain, such as actual processing times for orders, vehicle start times, and encountering traffic congestion during transportation. Therefore, employing optimization methods for uncertain environments in future research to address production and transportation coordination issues holds greater practical significance. (3) Enterprises typically need to optimize two or more conflicting objectives in real production and transportation scenarios. Hence, in the future, we mainly focus on the following tasks: (1) to apply Pareto-based multi-objective algorithms to solve scheduling problems in dynamic environments; and (2) to apply reinforcement learning algorithms and other efficient meta-heuristics[20–24] to solve the considered problems.

Data availability

The data that supports the findings of this study are available from the corresponding author, upon reasonable request.

Received: 14 August 2025; Accepted: 31 December 2025
Published online: 05 January 2026

References

1. Hou, Y., Wang, H., Fu, Y., Gao, K. & Zhang, H. Multi-objective brain storm optimization for integrated scheduling of distributed flow shop and distribution with maximal processing quality and minimal total weighted earliness and tardiness. *Comput. Ind. Eng.* **179**, 109217 (2023).
2. Bahmani, V., Adibi, M. A. & Mehdizadeh, E. Two-objective optimization for integrating parts ordering,two-stage assembly fow-shop scheduling, and distribution through routing. *Int. J. Industrial Eng.* **30**(3), 699 (2023).
3. Hou, Y., Fu, Y., Gao, K., Zhang, H. & Sadollah, A. Modelling and optimization of integrated distributed flow shop scheduling and distribution problems with time windows. *Expert Syst. Appl.* **187**, 115827 (2022).
4. He, P., Li, K. & Kumar, P. N. R. An enhanced branch-and-price algorithm for the integrated production and transportation scheduling problem. *Int. J. Prod. Res.* **60**(6), 1874–1889 (2022).
5. Zou, X., Liu, L., Li, K. & Li, W. A coordinated algorithm for integrated production scheduling and vehicle routing problem. *Int. J. Prod. Res.* **56**, 5005–5024 (2017).
6. Ganji, M., Kazemipoor, H., Hadji Molana, S. M. & Sajadi, S. M. A green multi-objective integrated scheduling of production and distribution with heterogeneous fleet vehicle routing and time windows. *J. Clean. Prod.* **259**, 120824 (2020).
7. Yin, Y., Wang, Y., Cheng, T. C. E., Wang, D. J. & Wu, C. C. Two-agent single-machine scheduling to minimize the batch delivery cost. *Comput. Ind. Eng.* **92**, 16–30 (2016).
8. Bachtenkirch, D. & Bock, S. Finding efficient make-to-order production and batch delivery schedules. *Eur. J. Oper. Res.* **297**(1), 133–152 (2022).
9. Guo, Z., Shi, L., Chen, L. & Liang, Y. A harmony search-based memetic optimization model for integrated production and transportation scheduling in Mto manufacturing. *Omega* **66**, 327–343 (2017).
10. Zuo, Y., Wang, P. & Li, M. A population diversity-based artificial bee colony algorithm for assembly hybrid flow shop scheduling with energy consumption. *Appl. Sci.* **13**(19), 10903 (2023).
11. Zhang, F., Pan, Y. & Yu, Y. An improved discrete artificial bee colony algorithm for distributed permutation flowshop scheduling problem with sequence dependent setup times, *2022 34th Chinese Control and Decision Conference (CCDC).* 4994–4999 (2022).
12. Zheng, Q. Q., Zhang, Y., He, L. J. & Tian, H. W. Discrete multi-objective artificial bee colony algorithm for green co-scheduling problem of ship lift and ship lock. *Adv. Eng. Inf.* **55**, 101897 (2023).
13. Haoran, L. I., Liang, G. A. O. & Xinyu, L. I. Discrete artificial bee colony algorithm for multi-objective distributed heterogeneous no-wait flowshop scheduling problem. *J. Mech. Eng.* **59**(2), 291–306 (2023).
14. Li, J. Q. et al. Hybrid artificial bee colony algorithm for a parallel batching distributed flow-shop problem with deteriorating jobs. *IEEE Trans. Cybern.* **50**, 2425–2439 (2020).
15. Johnson, S. M. Optimal two-and three-stage production schedules with setup times included. *Naval Res. Logistics Q.* **1**, 61–68 (1954).
16. Saber, R. G. & Ranjbar, M. Minimizing the total tardiness and the total carbon emissions in the permutation flow shop scheduling problem. *Comput. Oper. Res.* **138**, 105604 (2022).
17. Xu, X., Li, J., Zhou, M. & Yu, X. Precedence-constrained colored traveling salesman problem: an augmented variable neighborhood search approach. *IEEE Trans. Cybernetics*. **PP**, 1–12 (2021).
18. Cui, L., Liu, X., Lu, S. & Jia, Z. A variable neighborhood search approach for the resource-constrained multi-project collaborative scheduling problem. *Appl. Soft Comput.* **107**, 107480 (2021).
19. Sadati, M. E. H., Çatay, B. & Aksen, D. An efficient variable neighborhood search with Tabu shaking for a class of multi-depot vehicle routing problems. *Comput. Oper. Res.* **133**, 105269 (2021).
20. Chen, X. et al. *Optimizing Dynamic Flexible Job Shop Scheduling Using an Evolutionary multi-task Optimization Framework and Genetic Programming.* (IEEE Transactions on Evolutionary Computation, 2025).

21. Du, Y., Li, J., Li, C. & Duan, P. A reinforcement learning approach for flexible job shop scheduling problem with crane transportation and setup times. *IEEE Trans. Neural Netw. Learn. Syst.* **35**, 5695–5709. https://doi.org/10.1109/TNNLS.2022.3208942 (2024).
22. Wang, Z., Li, J., Chen, X., Duan, P. & Li, J. Uncertain interruptibility multiobjective flexible job shop via deep reinforcement learning based on heterogeneous graph Self-Attention. *IEEE Trans. Neural Networks Learn. Syst.* **2025** https://doi.org/10.1109/TNNLS.2025.3578368 (2025).
23. Du, Z. S. et al. Solving the permutation flow shop scheduling problem with sequence-dependent setup time via iterative greedy algorithm and imitation learning. *Math. Comput. Simul.* **234**, 169–193 (2025).
24. He, X., Pan, Q. K., Gao, L., Neufeld, J. S. & Gupta, J. N. D. Historical information based iterated greedy algorithm for distributed flowshop group scheduling problem with sequence-dependent setup times. *Omega* **123**, 102997 (2024).

Author contributions

Chenxin Dong: Conceptualization, Methodology, Idea, Software, Experiments, Writing - Original Draft, Visualization. Hui Zhang: Idea, Project administration, Supervision, Writing - review & editing, Validation.

Funding

This research is partially supported by Yunnan Key Laboratory of Modern Analytical Mathematics and Applications (No. 202302AN360007).

Declarations

Competing interests

The authors declare no competing interests.

Ethical approval

The authors declare that our research is ethical. This article does not contain any animal or human studies.

Additional information

Correspondence and requests for materials should be addressed to H.Z.