symmetry

MDPI

Article

A Shuffled Frog-Leaping Algorithm with Cooperations for Distributed Assembly Hybrid-Flow Shop Scheduling with Factory Eligibility

Deming Lei * and Tao Dai

School of Automation, Wuhan University of Technology, Wuhan 430070, China
* Correspondence: deminglei11@163.com or deminglei11@whut.edu.cn

Abstract: The distributed assembly scheduling problem with a hybrid-flow shop for fabrication is seldom studied, and some real-life constraints such as factory eligibility are seldom handled. In this study, a distributed assembly hybrid-flow shop-scheduling problem (DAHFSP) with factory eligibility is investigated, which has some symmetries on machines. A shuffled frog-leaping algorithm with cooperations (CSFLA) is applied to minimize makespan. A problem-related feature is used. Memeplexes are evaluated, and group 1, with the two best memeplexes, and group 2, with the two worst memeplexes, are formed. A new cooperation between memeplexes and an adaptive search strategy are implemented in groups 1 and 2, respectively. An adaptive cooperation between groups 1 and 2 is also given. Population shuffling is executed every T generations. A number of computational experiments are conducted. Computational results demonstrate that new strategies are effective and CSFLA is a very competitive algorithm for DAHFSP with factory eligibility.

Keywords: distributed assembly scheduling; hybrid-flow shop scheduling; factory eligibility; shuffled frog-leaping algorithm; cooperation

check for updates

Citation: Lei, D.; Dai, T. A Shuffled Frog-Leaping Algorithm with Cooperations for Distributed Assembly Hybrid-Flow Shop Scheduling with Factory Eligibility. *Symmetry* **2023**, *15*, 786. https://doi.org/10.3390/sym15040786

Academic Editor: Hsien-Chung Wu

Received: 22 February 2023
Revised: 17 March 2023
Accepted: 21 March 2023
Published: 23 March 2023

1. Introduction

The distributed hybrid-flow shop-scheduling problem (DHFSP) is the extended version of the hybrid-flow shop-scheduling problem (HFSP) in multiple factories, each of which has a hybrid-flow shop. In the past decade, DHFSP has attracted much attention and a number of results have been obtained [1–7].

DHFSP, with various processing constraints, has been extensively investigated, mainly by using heuristics and meta-heuristics, which include cooperative memetic algorithms [7], constructive heuristics [4], memetic algorithms [8], three fast heuristics and an adaptive genetic algorithm [9], hyper-heuristics [10], the iterative greedy (IG) algorithm [11], and the shuffled frog-leaping algorithm (SFLA) with Q-learning [12]. Energy efficiency, no wait, factory eligibility, third-party logistics, blocking, and assembly are dealt with.

Apart from the above constraints, DHFSP with multiprocessor tasks is solved by IG [13], dynamical SFLA [14], and evolutionary algorithms [15]. Some meta-heuristics are also applied to solve DHFSP with SDST, these algorithms are improved artificial bee colony (ABC) [16], discrete ABC [6], SFLA with memeplex quality [17], SFLA with memeplex grouping [18], and a diversified teaching–learning-based optimization [19].

In the past decades, distributed assembly scheduling problems (DASP), being composed of a fabrication stage with parallel machines or permutation flow shop, transportation stage, and assembly stage, have been extensively considered [20–25]; however, transportation is often neglected and DAHFSP, which is DASP with a hybrid-flow shop at the fabrication stage, is seldom studied.

As stated above, many works have been carried out on DHFSP with various real-life constraints such as no wait, blocking, SDST, and assembly in the past five years. DASP with constraints such as setup time and maintenance is also frequently studied; however, some

constraints, including factory eligibility, are seldom handled. Factory eligibility means that not all factories are eligible for each job. It is a typical constraint in multiple factories, which are obtained by mergers and acquisitions or given different production requirements. DHFSP with factory eligibility is also seldom investigated, let alone DAHFSP with factory eligibility. The considerations on factory eligibility will lead to high application value of the optimization results, thus, it is necessary to handle DAHFSP with factory eligibility.

SFLA is a meta-heuristic with a fast convergence speed, which models the behavior of frogs when searching for the location that has the most food [26]. It has been diffusely exploited to deal with different scheduling problems [14,17,18,27–36]; moreover, it can be found that SFLA has been successfully developed to solve DHFSP with SDST and multi-processor tasks and DAHFSP. The promising advantages and search abilities are proved; moreover, in recent years, some new optimization mechanisms, such as reinforcement learning and dynamical adjustment, have been adopted in SFLA and the performance is notably improved with the usage of new mechanisms. However, the works on SFLA with new mechanisms are very limited, and some mechanisms such as cooperation are seldom adopted in SFLA; thus, on the basis of the above analyses, it is concluded that SFLA with a new optimization mechanism may be a potential method to solve DAHFSP with factory eligibility.

In this study, DAHFSP with factory eligibility is considered and a new shuffled frog-leaping algorithm with cooperations (CSFLA) is presented to minimize makespan. A problem-related feature is used. Memeplexes are evaluated, and group 1 (with the two best memeplexes) and group 2 (with the two worst memeplexes) are formed. Two new cooperations between memeplexes and two adaptive search strategies are implemented. An adaptive cooperation between groups 1 and 2 is also given. Population shuffling is executed every T generations. A number of computational experiments are conducted. Computational results demonstrate that new strategies are effective and CSFLA is a very competitive algorithm for the considered DAHFSP.

The problem is depicted in Section 2. CSFLA for DAHFSP with factory eligibility is described in Section 3. Computational experiments on five algorithms are explored in Section 4. The conclusions and future topics are provided in the final section.

2. Problem Description

DAHFSP with factory eligibility is described as follows. There are n products, the component set of product i is Ψ_i according to bill of material. There are F heterogeneous factories, the factory f has a hybrid-flow shop, in which exists S component processing stages and m_l identical parallel machines at stage l as well as a machine TM_f for transportation and a machine AM_f for assembling. M_{flk} denotes the k-th processing machine at stage l in factory f.

All components of a product are first processed at the processing stage. When all components are processed, they are transported by TM_f to AM_f and the product is made. For a product i, all its components are handled as jobs in HFSP, and each component is fabricated in the same flow: stage 1, stage 2, ..., stage S.

com_{ij} denotes the jth component of product i and the processing time of com_{ij} is p_{jilf} at stage l in factory f. tr_{if} and as_{if} are the time for transportation and assembling of product i in factory f.

At least one product i has a set $\Theta_i \subset \{1, 2, \ldots, F\}$. Product i can only be fabricated, transported, and assembled in a factory belonging to Θ_i.

The constraints of DAHFSP have been introduced by Cai et al. [12].

The newly analyzed DAHFSP with factory eligibility is composed of factory assignment, HFSP for all components of each product, and product scheduling. Strong coupled relations are among these sub-problems.

For each product, the number of its components is limited. HFSP for all its components is a small-scale problem with makespan minimization and can be easily solved by heuris-

tics.This is a characteristic, so only factory assignment and scheduling for all products are required to be optimized.

The goal of the problem is to minimize makespan under the condition that all constraints are met. The optimization formulation is as follows:

$$\min C_{max} = \max\{C_i | i = 1, 2, \ldots, n\} \tag{1}$$

Subject to

$$\sum_{f=1}^{F} X_{jif} = 1, \forall j, i, f \in \Theta_i \tag{2}$$

$$X_{jif} = X_{j'if}, \forall i, f \in \Theta_i \tag{3}$$

$$\sum_{k=1}^{m_l} Y_{jiflk} = X_{jif}, \forall j, i, l, f \in \Theta_i \tag{4}$$

$$st_{ji1f} \geq 0, \forall j, i, f \in \Theta_i \tag{5}$$

$$st_{ji(l+1)f} \geq et_{jilf}, \forall j, i, l, f \in \Theta_i \tag{6}$$

$$et_{jilf} = st_{jilf} + p_{jilf}, \forall j, i, l, f \in \Theta_i \tag{7}$$

$$tst_{if} \geq \max\{et_{jiSf}\}, \forall j, i, f \in \Theta_i \tag{8}$$

$$tet_{if} = tst_{if} + tr_{if}, \forall i, f \in \Theta_i \tag{9}$$

$$ast_{if} \geq tet_{if}, \forall i, f \in \Theta_i \tag{10}$$

$$C_i = ast_{if} + as_{if}, \forall i, f \in \Theta_i \tag{11}$$

$$Z_{jj'ifl} + Z_{j'jifl} \leq 1, \forall j, j^{'}, i, l, f \in \Theta_i \tag{12}$$

$$Z_{jj'ifl} + Z_{j'jifl} \geq Y_{jiflk} + Y_{j'iflk} - 1, \forall f \in \Theta_i, l, i, j^{'} > j, k \in \{1, 2, \ldots, m_l\} \tag{13}$$

$$st_{jilf} \geq et_{jilf} - U \times (3 - Y_{jiflk} - Y_{j'iflk} - Z_{jj'ifl}), \forall j \neq j^{'}, i, l, k \in \{1, 2, \ldots, m_l\}, f \in \Theta_i \tag{14}$$

$$B_{ii'f} + B_{i'if} \leq 1, \forall i, i^{'}, f \in \Theta_i \tag{15}$$

$$B_{ii'f} + B_{i'if} \geq A_{if} + A_{i'f} - 1, \forall i, i^{'}, f \in \Theta_i \tag{16}$$

$$tst_{i'f} \geq tet_{if} - U \times (3 - A_{if} - A_{i'f} - B_{ii'f}), \forall i, i^{'}, f \in \Theta_i \tag{17}$$

$$D_{ii'f} + D_{i'if} \leq 1, \forall i, i^{'}, f \in \Theta_i \tag{18}$$

$$D_{ii'f} + D_{i'if} \geq E_{if} + E_{i'f} - 1, \forall i, i^{'}, f \in \Theta_i \tag{19}$$

$$ast_{i'f} \geq aet_{if} - U \times (3 - E_{if} - E_{i'f} - D_{ii'f}), \forall i, i^{'}, f \in \Theta_i \tag{20}$$

$$X_{jif} \in \{0,1\}, \forall j,i,f \in \Theta_i \tag{21}$$

$$Y_{jiflk} \in \{0,1\}, \forall j,i,l,k \in \{1,2,\ldots,m_l\}, f \in \Theta_i \tag{22}$$

$$Z_{jj'ifl} \in \{0,1\}, \forall j \neq j^{'},i,l,f \in \Theta_i \tag{23}$$

$$A_{if} \in \{0,1\}, \forall i,f \in \Theta_i \tag{24}$$

$$B_{ii'f} \in \{0,1\}, \forall i \neq i^{'},f \in \Theta_i \tag{25}$$

$$E_{if} \in \{0,1\}, \forall i,f \in \Theta_i \tag{26}$$

$$D_{ii'f} \in \{0,1\}, \forall i \neq i^{'},f \in \Theta_i \tag{27}$$

where C_{max} indicates maximum completion time of all products and C_i is the completion time of product i in formulation (1); constraint (2)–(7) are some constraints on components; constraint (8)–(11) are some constraints on products and constraint (12)–(14) show the constraints on machines for component fabrication; constraint (15)–(17) and (18)–(20), respectively, indicate the constraints on the transportation machine and assembly machine; constraints (21)–(27) show the decision variables. Abbreviations provide the notations and their descriptions.

For the problem with makespan, on each machine M_{flk} at each stage there exists a job-related symmetry, that is, two adjacent components are exchanged and the maximum completion time of all components on M_{flk} is not changed. The maximum completion time of the fabrication stage is also not varied; however, makespan may be changed after transportation and assembly when two adjacent components are exchanged on a M_{flk}.

Tables 1 and 2 show an illustrative example with five products and 2 factories, each of which has two stages for component fabrication. $\Theta_1 = \Theta_2 = \Theta_5 = \{1,2\}$, $\Theta_3 = \{1\}$, $\Theta_4 = \{2\}$. Figure 1 describes a schedule of the example.

Table 1. The example information of products.

Product i	$\lvert\Psi_i\rvert$	tr_{if}	as_{if}
		Factory 1: tr_{i1}**/Factory 2:** tr_{i2}	**Factory 1:** as_{i1}**/Factory 2:** as_{i2}
1	4	25/21	54/58
2	2	34/38	92/100
3	3	49/55	19/29
4	2	88/84	37/39
5	4	95/96	36/41

Table 2. The processing information of components.

Product i		**1**				**2**		**3**			**4**		**5**			
com_{ij}		**1**	**2**	**3**	**4**	**1**	**2**	**1**	**2**	**3**	**1**	**2**	**1**	**2**	**3**	**4**
Factory 1:	p_{ji11}	43	93	22	19	40	75	70	68	36	41	34	73	45	43	50
	p_{ji21}	54	83	47	96	34	41	43	100	95	49	25	69	57	78	33
Factory 2:	p_{ji12}	55	87	24	20	48	72	73	66	42	44	23	74	42	54	48
	p_{ji22}	46	91	48	92	27	39	49	94	99	42	34	65	42	69	45

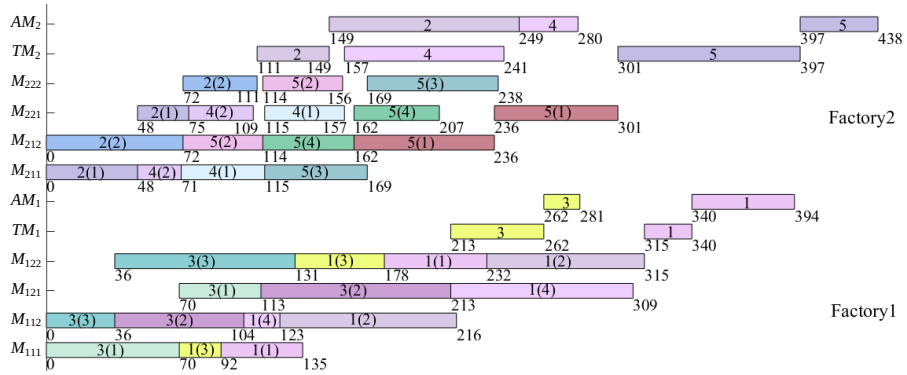

Figure 1. A schedule of the example.

3. CSFLA for DAHFSP with Factory Eligibility

A kind of cooperation is implemented between the best and the worst memeplex [27]. In this study, three cooperations are performed, two of which are between memeplexes and one of which is performed between two groups of memeplexes. The detailed steps of CSFLA are shown below.

3.1. Initialization

As analyzed above, only factory assignment and product scheduling are needed to be optimized after HFSP, for all components of each product are solved by heuristics.For DAHFSP with factory eligibility, n products and F factories, its solution consists of a factory assignment string $[\theta_1, \theta_2, \ldots, \theta_n]$ and a product scheduling string $[\pi_1, \pi_2, \ldots, \pi_n]$, where $\theta_i \in \Theta_i$, $\pi_i \in \{1, 2, \ldots, n\}$.

Algorithm 1 shows the detailed steps of decoding, where if more than one machine has the same smallest available time in line 8, a machine with the smallest subscript is chosen.

Algorithm 1 Decoding

1: **for** $f = 1$ to F **do**
2: decide all assigned products by factory assignment string and obtain a permutation of these products in factory f, suppose that the permutation is $\pi_1, \pi_2, \ldots, \pi_g$
3: **for** $l = 1$ to g **do**
4: determine a permutation of all components for product π_l by a heuristic
5: suppose the permutation is $com_{\pi_l 1}, \ldots, com_{\pi_l |\Psi_{\pi_l}|}$,
6: **for** $h = 1$ to $|\Psi_{\pi_l}|$ **do**
7: **for** $j = 1$ to S **do**
8: fabricate component $com_{\pi_l h}$ on a machine M_{fjk} with the smallest available time
9: **end for**
10: **end for**
11: move all components of product π_l by TM_f to AM_f and assemble all components.
12: **end for**
13: **end for**

The heuristic for a permutation of all components of product i is described below. For each component com_{ij}, compute $\sum_{l=1}^{S} p_{jilf}$; sort all components of product i in the ascending order of $\sum_{l=1}^{S} p_{jilf}$ and obtain a permutation.

For the example in Tables 1 and 2, a solution is represented as $[1, 2, 1, 2, 2]$ and $[2, 4, 3, 5, 1]$. In factory 2, products 2, 4, 5 are assigned and their coresponding permutation is 2, 4, 5. Product 2 is first handled; for product 2, a permutation of its components is 1, 2, for component 1 of product 2, M_{211} and M_{221} are chosen. The final schedule is shown in Figure 1.

An initial population P with N initial solutions is randomly generated and then divided into s memeplexes $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_s$ according to the descriptions in [26]. $N = s \times Msize$, where $Msize$ is the size of the memeplex.

Quality Me_i of memeplex $\mathcal{M}_i$ is evaluated by

$$Me_i = \sum_{x \in \mathcal{M}_i} \left| \left\{ y \in P \middle| C^y_{max} > C^x_{max} \right\} \right| \tag{28}$$

All memeplexes are sorted in the descending order of Me_i; suppose that $Me_1 \geq Me_2 \geq \ldots \geq Me_s$. Group 1 is composed of $\mathcal{M}_1, \mathcal{M}_2$ and group 2 is made up of $\mathcal{M}_s, \mathcal{M}_{s-1}$. A cooperation is applied in each group and cooperation between two groups is implemented.

3.2. Two Cooperations in the Search Process of Group 1

In the search process of group 1, the memeplex search is implemented by a cooperation between $\mathcal{M}_1, \mathcal{M}_2$ in group 1, then, to implement cooperation between groups 1 and 2, the number η of the reinforcement search in group 1 is computed and the reinforcement searches are executed in $\mathcal{M}_1, \mathcal{M}_2$, respectively. Algorithm 2 shows the cooperation-based search process of $\mathcal{M}_1, \mathcal{M}_2$ and Algorithm 3 describes the search process of group 1, where Ω is used to store some best solutions and Ψ is applied to keep historical data, and the integer V means the volume of Ω and Ψ. The initial Ω consists of the best V solutions of population P and the initial Ψ is empty. Ω is updated by a new solution that is better than the worst solution in it, and Ψ has the same update as with Ω if it is accumulated fully by solutions. NS_l denotes neighborhood search, $l = 1, 2, 3$.

$$\eta = 2 \times \mu \times (Me_1 + Me_2)/(Me_1 + Me_2 + Me_{s-1} + Me_s) \tag{29}$$

Algorithm 2 Cooperation within group 1

1: **for** $t = 1$ to $2 \times \mu$ **do**
2: decide a set $\Delta_i = \left\{ x \in \mathcal{M}_i \middle| C^x_{max} > \sum_{y \in \mathcal{M}_i} C^y_{max} \middle/ Msize \right\}, i = 1, 2$
3: randomly choose $x \in \mathcal{M}_1 \backslash \Delta_1$ and $y \in \mathcal{M}_2 \backslash \Delta_2$ and choose a better solution from x, y as optimization object, suppose x is chosen, $\gamma = 0$
4: execute global search between x and y and obtain a new z
5: **if** $C^z_{max} < C^x_{max}$ **then**
6: x is replaced and Ω is updated with z, $\gamma = 1$
7: **else**
8: **if** $C^z_{max} < C^y_{max}$ **then**
9: y is replaced and Ω is updated with z, $\gamma = 1$
10: **else**
11: z is used to renew memory Ψ
12: **end if**
13: **end if**
14: **if** $\gamma = 0$ **then**
15: perform global search between x, $gbest$, generate a new z and execute lines 5–13
16: **end if**
17: **if** $\gamma = 0$ **then**
18: execute NS_1 on x
19: **end if**
20: **end for**

$$\lambda^x_i = |C^x_{max} - \sum_{y \in \mathcal{M}_i} C^y_{max} \Big/ Msize| \Big/ \sum_{y \in \mathcal{M}_i} C^y_{max} \Big/ Msize \tag{30}$$

Cai and Lei [27] proposed two global search operators for a product scheduling string and factory assignment string. The global search is depicted as follows. For solutions x, y, randomly choose one of two operators with the same probability, and produce a new solution by the chosen operator.

Eight neighborhood structures are used. $\mathcal{N}_1$ is shown below. Randomly decide a factory f and stochastically select products i, j assigned in factory f; suppose that $\pi_g = j$, insert product i into the position g of scheduling string. $\mathcal{N}_2$ is described as follows. Randomly choose a factory f and a product i in factory f, stochastically decide a factory l, if $l \in \Theta_i$, then let $\theta_i = l$; otherwise, randomly choose a factory v, if $v \in \Theta_i$, then let $\theta_i = v$. $\mathcal{N}_3$ is similar to $\mathcal{N}_1$; products i, j are swapped in scheduling string. $\mathcal{N}_4$ has similar steps to $\mathcal{N}_2$; factories f, l are randomly decided, products i, j are randomly chosen from factories f, l, respectively, then i, j are exchanged in the scheduling string. Let $\theta_i = v$ and $\theta_j = f$ if $f \in \Theta_j$, $v \in \Theta_i$; if $f \in \Theta_j$ or $v \in \Theta_i$ are not met, randomly choose a product j' from factory v, if $f \in \Theta_{j'}$, $v \in \Theta_i$, then i, j' are exchanged in the scheduling string and let $\theta_i = v$ and $\theta_{j'} = f$.

Algorithm 3 Reinforcement search process in group 1

1: execute cooperation-based search process of $\mathcal{M}_1, \mathcal{M}_2$ in Algorithm 2
2: compute $u_1 = \eta \times Me_2/(Me_1 + Me_2)$ and $u_2 = \eta - u_1$
3: **for** $i = 1$ to 2 **do**
4: **for** $v = 1$ to u_i **do**
5: decide sets Δ_i, randomly choose a $x \in \Delta_i, x \neq x_w$ and compute λ_i^x
6: **if** $\lambda_i^x \leq |\Delta_i|/Msize$ **then**
7: $\gamma = 0$
8: **if** Ψ is not empty **then**
9: randomly select a $y \in \Psi$, replace x with y if $C_{max}^y < C_{max}^x$
10: **if** $C_{max}^y \geq C_{max}^x$ **then**
11: execute global search between y and $gbest$, obtain a new z and replace x with z, $\gamma = 1$ if $C_{max}^z < C_{max}^x$
12: **end if**
13: **end if**
14: **if** $\gamma = 0$ or Ψ is empty **then**
15: randomly choose a $y \in \Omega$, perform global search between x, y, produce a new z', if z' is better than x, $x = z'$, otherwise, perform NS_1 on y, $x = y$
16: **end if**
17: **else**
18: perform global search between $x, gbest$ and obtain a new z, if $C_{max}^z < C_{max}^x$, then replace x with z; otherwise, randomly choose a $y \in \Omega$, select one of NS_2 and NS_3 with the same probability, execute the chosen one on y and y substitutes for x
19: **end if**
20: **end for**
21: **end for**

Suppose that factory 1 has the biggest completion time and factory 2 has the smallest completion time. When $\mathcal{N}_1, \mathcal{N}_3$ are performed in factory 1, then $\mathcal{N}_5$ and $\mathcal{N}_7$ are obtained. $\mathcal{N}_6$ is described as follows. Randomly select a product i in factory 1 and a product j from factory 2, if $2 \in \Theta_i$, then let $\theta_i = 2$; otherwise, randomly decide a product j from factory 1, if $2 \in \Theta_j$, then let $\theta_j = 2$. $\mathcal{N}_8$ is shown below. Products i, j are randomly chosen from factories $1, 2$, respectively, i, j are exchanged in the scheduling string, and let $\theta_i = 2$ and $\theta_j = 1$ if $1 \in \Theta_j$ and $2 \in \Theta_i$.

NS_1 for solution x is shown as follows. Randomly choose a $v \in \{1, 2, 3, 4\}$, sequentially execute $\mathcal{N}_{2\times(v-1)+1}, \mathcal{N}_{2\times(v-1)+2}$ on x, a new solution z is obtained, if z is better than x, then z substitutes for x.

NS_2 is described below. Define $\alpha_1 = 2, \alpha_2 = 1, \alpha_3 = 4, \alpha_4 = 3$, let $g = 1$, $\gamma = 0$, repeat the following steps until $\gamma > 0$ or $g > 4$: produce a solution $z \in \mathcal{N}_{\alpha_g}(x)$, if z is better than x, then z substitutes for x and $\gamma = 1$; otherwise, $g = g + 1$, where $\mathcal{N}_g(x)$ is the set of neighborhood solutions produced by $\mathcal{N}_g$ on x.

NS_3 has the same steps as NS_2. In NS_2, $\alpha_1 = 6, \alpha_2 = 5, \alpha_3 = 8, \alpha_4 = 7$.

In Algorithm 3, for each $x \in \Delta_i$, C_{max}^x is greater than the average makespan of $\mathcal{M}_i, i = 1, 2$, if $\lambda_i^x = 0.1(0.6)$, then $C_{max}^x = 1.1 \sum_{y\in\mathcal{M}_i} C_{max}^y \Big/ Msize$ $(1.6 \sum_{y\in\mathcal{M}_i} C_{max}^y \Big/ Msize)$, the condition of line 6 means that C_{max}^x slightly exceeds the average makespan and the condition of line 17 denotes that C_{max}^x exceeds greatly the average makespan. The search operator is decided adaptively by one of the above conditions; moreover, the cooperation-based search process

between $\mathcal{M}_1, \mathcal{M}_2$ acts on solutions out of Δ_1, Δ_2 and reinforcement search is performed on solutions in Δ_1, Δ_2, as a result, exploration ability of CSFLA is intensified greatly.

3.3. Cooperation-Based Search Process of Group 2

Algorithm 4 describes the cooperation-based search process of group 2, where δ and num are defined.

$$\delta = \frac{Me_1 + Me_2 - Me_{s-1} - Me_s}{Me_1 + Me_2} \tag{31}$$

$$num = \max\{|\Phi|, 2|\Phi| \times \delta\} \tag{32}$$

Algorithm 4 Cooperation-based search process of group 2

1: let the set $TE = \mathcal{M}_{s-1} \cup \mathcal{M}_s$;
2: decide $\Phi = \left\{ x \in TE \middle| C^x_{max} > \sum_{y \in TE} C^y_{max} \middle/ (2 \times Msize) \right\}$
3: **for** $t = 1$ to $2\mu - \eta$ **do**
4: decide two best solutions $x_1, x_2 \in TE$, suppose x_1 is chosen, $C^{x_1}_{max} < C^{x_2}_{max}$, a randomly chose solution $x_3 \in \Omega$
5: **if** $\delta \leq \frac{|\Phi|}{2 \times Msize}$ **then**
6: randomly choose one of x_1, x_2,let $\gamma = 0$
7: **if** Ψ is not empty **then**
8: randomly select $y \in \Psi$, if y is better than x_1, then $x_1 = y$, $\gamma = 1$; otherwise, if y is better than x_2, then $x_2 = y$ and $\gamma = 1$
9: **end if**
10: **if** Ψ is empty or $\gamma = 0$ **then**
11: execute similar steps with lines 4-18 of Algorithm 2
12: **end if**
13: **else**
14: perform global search between x_3, $gbest$ and obtain a new z
15: **if** $C^z_{max} < C^{x_3}_{max}$ **then**
16: $x_1 = x_3$ and $x_3 = z$
17: **else**
18: let $\zeta = 0$; if $C^z_{max} < C^{x_1}_{max}$, let $x_1 = z$, $\zeta = 1$; if $C^z_{max} < C^{x_2}_{max}$, let $x_2 = z$, $\zeta = 1$
19: **if** $\zeta = 0$ **then**
20: execute a chosen one from NS_2 and NS_3 with the same probability on x_3
21: **end if**
22: **end if**
23: **end if**
24: **end for**
25: **for** $r = 1$ to num **do**
26: **if** $rand < |\Phi|/(2 \times Msize)$ **then**
27: randomly choose $x \in \mathcal{M}_1$ and replace the worst solution in TE with x
28: **else**
29: **if** $rand > |\Phi|/(2 \times Msize)$ **then**
30: randomly choose $x \in \mathcal{M}_2$ and replace the worst solution in TE with x
31: **else**
32: randomly choose $x \in \Omega$ and replace the worst solution in TE with x
33: **end if**
34: **end if**
35: choose one NS_i with the same probability from NS_1, NS_2, NS_3 and execute the chosen NS_i on the replaced solution
36: **end for**

The similar steps with lines 4–18 of Algorithm 2 are shown below. x is optimization object, which is randomly chosen from x_1, x_2, y is x_3 in all lines except lines 5–9, in which x, y is changed into x_1, x_2, respectively.

In Algorithm 4, $\mathcal{M}_s$ and $\mathcal{M}_{s-1}$ are combined into a new memeplex TE and search process in TE is executed by an adaptive search strategy based on δ. The search times of group 2 is $2\mu - \eta$, the reinforcement search of group 1 is executed η times in Algorithm 3, and $\mathcal{M}_s$, $\mathcal{M}_{s-1}$ are updated by using solutions from $\mathcal{M}_1$, $\mathcal{M}_2$ or Ω. This is an adaptive cooperation between two groups.

3.4. Algorithm Description

Algorithm 5 describes the detailed steps of CSFLA, where T is the integer.

Algorithm 5 CSFLA

randomly produce initial population P, let $gen = 1$
divide population P into s memeplexes
while stopping condition is not met **do**
  compute Me_i for each memeplex $\mathcal{M}_i$ and construct groups 1 and 2
  execute cooperation within group 1 and reinforcement search process in group 1
  perform search process of each memeplex $\mathcal{M}_i, i \neq 1, 2, s-1, s$
  execute cooperation-based search process of group 2
  if gen is exactly divided by T **then**
    perform memeplex shuffling and divide population P into s memeplexes
  end if
  $gen = gen + 1$
end while

The search process of $\mathcal{M}_i, i \neq 1, 2, s-1, s$ is depicted below. $x_b \in \mathcal{M}_i$ is the optimization object, a solution $y \in \Omega$ is randomly decided, a global search is performed between x_b and y, if $C^z_{max} < C^{x_b}_{max}$, then $x_b = z$; otherwise, execute a global search between $x_b, gbest$ and produce a new z', if $C^{z'}_{max} < C^{x_b}_{max}$, then $x_b = z'$; otherwise, perform a chosen one of NS_1, NS_2, NS_3 with the same probability on x_b.

When gen is divided exactly by T, all evolved memeplexes are formed into a new population P.

Unlike the previous SFLA [14,17,18,27,35], CSFLA has some new features. (1) Memeplexes are evaluated and two groups are formed by using $\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_{s-1}, \mathcal{M}_s$. (2) A cooperation between two memeplexes is performed in groups 1 and 2, respectively, and an adaptive cooperation between groups 1 1 and 2 is implemented, so three cooperations are used. (3) An adaptive search strategy is applied in groups 1 and 2, respectively, and population shuffling is executed every T generations; as a result, memeplexes and groups can exist in T generations and can be evolved well by using the same solution structure in each memeplex.

4. Computational Experiments

All experiments are implemented by using Microsoft Visual C++ 2019 and run on 8.0G RAM 2.4 GHz CPU PC.

4.1. Test Instances and Comparative Algorithms

112 instances are used, each of which is depicted as $n \times F \times S$. $p_{jilf} \in [1, 100]$, $tr_{if} \in [1, 100]$, $as_{if} \in [1, 100]$, $|\Psi_i| \in [2, 5]$, $m_l \in [2, 5]$. All of the above data are integers.

A hybrid variable neighborhood search (HVNS) [37], improved discrete cuckoo optimization algorithm (IDCOA) [38], and improved whale optimization algorithm (IWOA) [39] are chosen as comparative algorithms.

HVNS is used to solve the distributed assembly flow shop-scheduling problem. HVNS can be applied to address DAHFSP after heuristics for HFSP for all components of each product in Section 3.1 is used. IWOA is to solve distributed assembly flow shop scheduling with transportation. It can be used to solve our DAHFSP after the heuristic for HFSP of Section 3.1 is adopted.

IDCOA with only product permutation is handled to deal with the three-stage assembly flow shop-scheduling problem. To solve DAHFSP, heuristics for HFSP for all components of each product is added, factory assignment and $\mathcal{N}_5$ are adopted, and a global search of CSFLA substitutes for the immigration operator of IDCOA is carried out.

SFLA is used to test the effect of the new strategies in CSFLA. The search process within each memeplex has similar steps with the search process of $\mathcal{M}_3$ SFLA, however, y is the second best solution in $\mathcal{M}_i$ and the random way of Section 3 substitutes for the random selection from NS_1, NS_2, NS_3, no cooperations are used and T is 1.

4.2. Parameter Settings

CSFLA has following parameters: N, s, μ, T, V, and a stopping condition. It can be found that CSFLA can converge well when time reaches $0.1 \times n \times S$ seconds CPU; moreover, when $0.1 \times n \times S$ seconds CPU time is applied, all comparative algorithms also converge well, so this time is chosen as stopping condition.

The taguchi method [40] is used to decide the settings for other parameters by using the instance $60 \times 4 \times 4$. Table 3 shows the levels of each parameter. The orthogonal array $L_{16}(4^5)$ is tested. CSFLA with each combination runs 10 times independently, for instance $60 \times 4 \times 4$.

Table 3. Parameters and their levels.

Parameters	Factor Level			
	1	**2**	**3**	**4**
N	60	90	120	150
s	5	6	10	15
μ	30	40	50	60
T	2	3	4	5
V	3	4	5	6

Figure 2 shows the results of MIN and S/N ratio, which is defined as $-10 \times \log_{10}(MIN^2)$. It can be found from Figure 2 that CSFLA with following combination $N = 60$, $s = 10$, $\mu = 50$, $T = 5$, $V = 4$ can obtain better results than CSFLA with other combinations, so the above parameter settings are chosen.

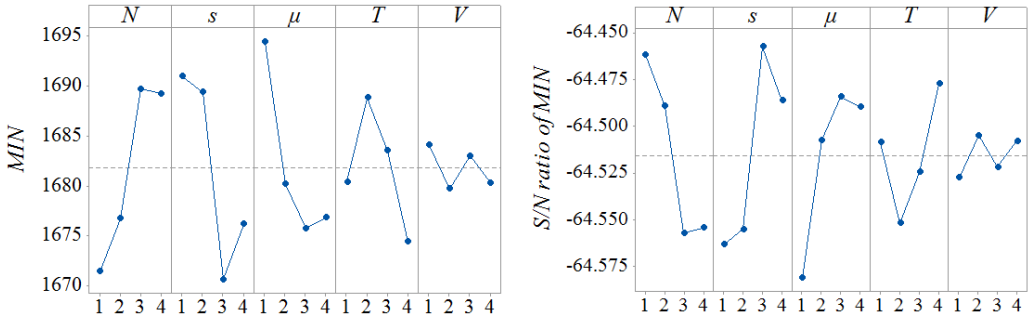

Figure 2. Main effect plot for mean MIN and S/N ratio.

SFLA has $N = 60$, $s = 10$, $\mu = 50$, and the above stopping condition.

The parameter settings of the three comparative algorithms are directly selected from the references, except for the stopping condition, because of the effectiveness of these settings of each comparative algorithm.

4.3. Results and Discussions

CSLFA, SFLA, and three comparative algorithms are compared. Each algorithm randomly runs 10 times for each instance. MIN (MAX) denotes the best (worst) solution's

C_{max} found in the 10 runtimes and AVG denotes the average C_{max} of solutions in 10 runtimes. Tables 4–6 describe the corresponding results of five algorithms. Figures 3 and 4 show convergence curves and box plots of all algorithms. The relative percentage deviation (RPD) between the best-performing algorithm and the other four algorithms is used in Figure 4 and Table 7 gives the results of the paired-sample Wilcoxon test.

Table 4. Computational results of instances of five algorithms on *MIN*.

Instance	CSFLA	SFLA	IDCOA	HVNS	IWOA	Instance	CSFLA	SFLA	IDCOA	HVNS	IWOA
20 × 2 × 2	**664.0**	722.0	689.0	699.0	702.0	60 × 4 × 2	**1033.0**	1148.0	1055.0	1078.0	1098.0
20 × 2 × 4	**1071.0**	1161.0	1099.0	1105.0	1103.0	60 × 4 × 4	**1651.0**	1798.0	1687.0	1719.0	1757.0
20 × 2 × 6	**1351.0**	1418.0	1372.0	1377.0	1399.0	60 × 4 × 6	**1229.0**	1381.0	1264.0	1319.0	1338.0
20 × 2 × 8	**1499.0**	1639.0	1537.0	1563.0	1574.0	60 × 4 × 8	**1705.0**	1840.0	1730.0	1760.0	1764.0
20 × 3 × 2	**711.0**	769.0	730.0	741.0	765.0	60 × 5 × 2	**1172.0**	1274.0	1186.0	1284.0	1260.0
20 × 3 × 4	**903.0**	994.0	932.0	922.0	953.0	60 × 5 × 4	**1271.0**	1399.0	1292.0	1313.0	1344.0
20 × 3 × 6	**846.0**	904.0	860.0	881.0	907.0	60 × 5 × 6	**1483.0**	1650.0	1537.0	1579.0	1628.0
20 × 3 × 8	**1158.0**	1245.0	1179.0	1174.0	1223.0	60 × 5 × 8	**1609.0**	1745.0	1619.0	1766.0	1706.0
20 × 4 × 2	**405.0**	455.0	414.0	426.0	434.0	80 × 2 × 2	**3846.0**	3915.0	3850.0	3895.0	3872.0
20 × 4 × 4	**579.0**	670.0	603.0	616.0	649.0	80 × 2 × 4	**2624.0**	2764.0	2627.0	2674.0	2719.0
20 × 4 × 6	**816.0**	872.0	841.0	849.0	844.0	80 × 2 × 6	**3715.0**	3910.0	3741.0	3767.0	3855.0
20 × 4 × 8	**916.0**	1001.0	917.0	946.0	969.0	80 × 2 × 8	**4184.0**	4421.0	4230.0	4247.0	4372.0
20 × 5 × 2	**455.0**	526.0	476.0	499.0	520.0	80 × 3 × 2	**1446.0**	1572.0	1480.0	1476.0	1537.0
20 × 5 × 4	**608.0**	694.0	617.0	668.0	659.0	80 × 3 × 4	**1548.0**	1675.0	1556.0	1611.0	1641.0
20 × 5 × 6	**808.0**	898.0	814.0	849.0	876.0	80 × 3 × 6	**2603.0**	2715.0	2610.0	2666.0	2690.0
20 × 5 × 8	**874.0**	955.0	882.0	918.0	908.0	80 × 3 × 8	**2753.0**	2933.0	2782.0	2800.0	2818.0
40 × 2 × 2	**1356.0**	1423.0	1362.0	1370.0	1407.0	80 × 4 × 2	**1885.0**	1944.0	1895.0	1962.0	1939.0
40 × 2 × 4	**1887.0**	1982.0	1902.0	1913.0	1950.0	80 × 4 × 4	**1994.0**	2155.0	2035.0	2093.0	2083.0
40 × 2 × 6	**1629.0**	1775.0	1655.0	1655.0	1737.0	80 × 4 × 6	**2203.0**	2437.0	2261.0	2324.0	2295.0
40 × 2 × 8	**2285.0**	2395.0	2323.0	2323.0	2380.0	80 × 4 × 8	**2230.0**	2388.0	2253.0	2372.0	2361.0
40 × 3 × 2	**930.0**	1031.0	951.0	944.0	961.0	80 × 5 × 2	**978.0**	1120.0	1010.0	1028.0	1067.0
40 × 3 × 4	**1427.0**	1562.0	1456.0	1473.0	1502.0	80 × 5 × 4	**1703.0**	1864.0	1739.0	1782.0	1799.0
40 × 3 × 6	**1568.0**	1695.0	1583.0	1592.0	1626.0	80 × 5 × 6	**1747.0**	1946.0	1808.0	1886.0	1881.0
40 × 3 × 8	**1564.0**	1691.0	1571.0	1600.0	1623.0	80 × 5 × 8	**1826.0**	2011.0	1850.0	1942.0	1930.0
40 × 4 × 2	**885.0**	979.0	919.0	926.0	939.0	100 × 2 × 2	**3080.0**	3135.0	3086.0	3117.0	3136.0
40 × 4 × 4	**1137.0**	1219.0	1142.0	1188.0	1202.0	100 × 2 × 4	4874.0	4970.0	**4866.0**	4929.0	4936.0
40 × 4 × 6	**1184.0**	1282.0	1218.0	1239.0	1258.0	100 × 2 × 6	**4711.0**	4870.0	4755.0	4819.0	4822.0
40 × 4 × 8	**1336.0**	1497.0	1384.0	1428.0	1420.0	100 × 2 × 8	**4866.0**	5075.0	4891.0	4913.0	4986.0
40 × 5 × 2	**635.0**	724.0	673.0	690.0	706.0	100 × 3 × 2	**2074.0**	2226.0	2103.0	2105.0	2148.0
40 × 5 × 4	**934.0**	1084.0	966.0	973.0	1026.0	100 × 3 × 4	**3189.0**	3375.0	3216.0	3288.0	3287.0
40 × 5 × 6	**1122.0**	1283.0	1160.0	1162.0	1218.0	100 × 3 × 6	**3035.0**	3173.0	3070.0	3126.0	3101.0
40 × 5 × 8	**1299.0**	1462.0	1327.0	1332.0	1387.0	100 × 3 × 8	**3591.0**	3809.0	3664.0	3710.0	3752.0
60 × 2 × 2	**1646.0**	1749.0	1657.0	1676.0	1702.0	100 × 4 × 2	**1613.0**	1733.0	1650.0	1698.0	1670.0
60 × 2 × 4	**2808.0**	2920.0	2824.0	2863.0	2890.0	100 × 4 × 4	**2568.0**	2724.0	2580.0	2613.0	2647.0
60 × 2 × 6	**3027.0**	3195.0	3037.0	3065.0	3170.0	100 × 4 × 6	**2676.0**	2870.0	2709.0	2773.0	2772.0
60 × 2 × 8	**2863.0**	3001.0	2891.0	2927.0	2941.0	100 × 4 × 8	**2526.0**	2733.0	2572.0	2686.0	2693.0
60 × 3 × 2	**2015.0**	2096.0	2023.0	2047.0	2054.0	100 × 5 × 2	**1869.0**	2027.0	1880.0	1965.0	1889.0
60 × 3 × 4	**2099.0**	2278.0	2126.0	2148.0	2215.0	100 × 5 × 4	**1374.0**	1466.0	1399.0	1449.0	1419.0
60 × 3 × 6	**2211.0**	2339.0	2222.0	2273.0	2294.0	100 × 5 × 6	**1684.0**	1851.0	1738.0	1862.0	1765.0
60 × 3 × 8	**2331.0**	2457.0	2344.0	2378.0	2403.0	100 × 5 × 8	**2205.0**	2434.0	2252.0	2365.0	2312.0
120 × 2 × 2	**3422.0**	3479.0	3463.0	3433.0	3476.0	140 × 2 × 2	**4265.0**	4358.0	4364.0	4271.0	4321.0
120 × 2 × 4	**5671.0**	5720.0	5701.0	5683.0	5694.0	140 × 2 × 4	**6415.0**	6504.0	6590.0	6425.0	6507.0
120 × 2 × 6	**5951.0**	6040.0	6043.0	5954.0	6031.0	140 × 2 × 6	**6614.0**	6706.0	6677.0	6625.0	6665.0
120 × 2 × 8	**5704.0**	5787.0	5775.0	5712.0	5788.0	140 × 2 × 8	**6533.0**	6610.0	6600.0	6535.0	6622.0
120 × 3 × 2	2229.0	2253.0	2263.0	**2202.0**	2238.0	140 × 3 × 2	2563.0	2619.0	2609.0	**2557.0**	2600.0
120 × 3 × 4	**2411.0**	2496.0	2489.0	2444.0	2483.0	140 × 3 × 4	2698.0	2757.0	2773.0	**2673.0**	2758.0
120 × 3 × 6	3846.0	3997.0	3941.0	**3802.0**	3911.0	140 × 3 × 6	**4638.0**	4664.0	4690.0	4655.0	4673.0
120 × 3 × 8	**4265.0**	4428.0	4382.0	4272.0	4381.0	140 × 3 × 8	**3438.0**	3530.0	3569.0	3451.0	3526.0
120 × 4 × 2	**1741.0**	1785.0	1747.0	1771.0	1767.0	140 × 4 × 2	2002.0	2016.0	2035.0	**1964.0**	1988.0
120 × 4 × 4	**2145.0**	2238.0	2195.0	2146.0	2203.0	140 × 4 × 4	**2425.0**	2482.0	2433.0	2486.0	2481.0
120 × 4 × 6	**2258.0**	2378.0	2376.0	2272.0	2383.0	140 × 4 × 6	**3517.0**	3562.0	3619.0	3573.0	3588.0
120 × 4 × 8	**3146.0**	3209.0	3184.0	3150.0	3151.0	140 × 4 × 8	**3839.0**	3937.0	3912.0	3842.0	3915.0
120 × 5 × 2	**2317.0**	2358.0	2363.0	2329.0	2350.0	140 × 5 × 2	**1904.0**	1941.0	1915.0	1959.0	1925.0
120 × 5 × 4	**2558.0**	2606.0	2581.0	2616.0	2633.0	140 × 5 × 4	**2875.0**	2949.0	2954.0	2882.0	2952.0
120 × 5 × 6	2601.0	2608.0	2651.0	**2585.0**	2648.0	140 × 5 × 6	**3142.0**	3297.0	3198.0	3315.0	3211.0
120 × 5 × 8	**2655.0**	2690.0	2724.0	2747.0	2729.0	140 × 5 × 8	3245.0	3405.0	3359.0	**3199.0**	3317.0

Table 5. Computational results of instances of five algorithms on *MAX*.

Instance	CSFLA	SFLA	IDCOA	HVNS	IWOA	Instance	CSFLA	SFLA	IDCOA	HVNS	IWOA
20 × 2 × 2	**691.0**	790.0	726.0	760.0	824.0	60 × 4 × 2	**1064.0**	1218.0	1085.0	1238.0	1181.0
20 × 2 × 4	**1099.0**	1205.0	1124.0	1150.0	1225.0	60 × 4 × 4	**1692.0**	1923.0	1723.0	2223.0	1834.0
20 × 2 × 6	**1374.0**	1470.0	1414.0	1430.0	1458.0	60 × 4 × 6	**1285.0**	1452.0	1311.0	1495.0	1462.0
20 × 2 × 8	**1554.0**	1754.0	1600.0	1642.0	1735.0	60 × 4 × 8	**1754.0**	1924.0	1773.0	2236.0	1920.0
20 × 3 × 2	**728.0**	857.0	758.0	853.0	815.0	60 × 5 × 2	**1239.0**	1448.0	1241.0	1551.0	1304.0
20 × 3 × 4	**938.0**	1072.0	978.0	1116.0	1113.0	60 × 5 × 4	**1289.0**	1514.0	1328.0	1789.0	1451.0
20 × 3 × 6	**895.0**	1000.0	906.0	1054.0	1047.0	60 × 5 × 6	**1532.0**	1809.0	1568.0	2073.0	1725.0
20 × 3 × 8	**1192.0**	1354.0	1236.0	1285.0	1350.0	60 × 5 × 8	**1663.0**	1886.0	1698.0	1997.0	1807.0
20 × 4 × 2	**425.0**	493.0	438.0	476.0	465.0	80 × 2 × 2	**3869.0**	3986.0	3892.0	3960.0	4006.0
20 × 4 × 4	**611.0**	714.0	640.0	665.0	758.0	80 × 2 × 4	**2655.0**	2844.0	2691.0	2770.0	2818.0
20 × 4 × 6	**844.0**	1004.0	872.0	1020.0	1002.0	80 × 2 × 6	**3765.0**	3987.0	3791.0	3944.0	3953.0
20 × 4 × 8	**954.0**	1082.0	959.0	1103.0	1051.0	80 × 2 × 8	**4262.0**	4574.0	4295.0	4496.0	4508.0
20 × 5 × 2	**487.0**	611.0	523.0	672.0	571.0	80 × 3 × 2	**1482.0**	1674.0	1520.0	1784.0	1613.0
20 × 5 × 4	**633.0**	750.0	650.0	889.0	734.0	80 × 3 × 4	**1581.0**	1774.0	1610.0	1740.0	1727.0
20 × 5 × 6	**838.0**	988.0	890.0	1037.0	991.0	80 × 3 × 6	**2648.0**	2836.0	2664.0	2846.0	2740.0
20 × 5 × 8	**909.0**	1031.0	935.0	1090.0	1052.0	80 × 3 × 8	**2817.0**	3031.0	2843.0	2894.0	2947.0
40 × 2 × 2	**1372.0**	1483.0	1391.0	1471.0	1533.0	80 × 4 × 2	**1909.0**	2102.0	1942.0	2425.0	2124.0
40 × 2 × 4	**1922.0**	2061.0	1930.0	2013.0	2046.0	80 × 4 × 4	**2029.0**	2272.0	2061.0	2518.0	2197.0
40 × 2 × 6	1696.0	1860.0	**1695.0**	1737.0	1803.0	80 × 4 × 6	**2273.0**	2517.0	2303.0	2511.0	2457.0
40 × 2 × 8	**2350.0**	2497.0	2358.0	2421.0	2460.0	80 × 4 × 8	**2279.0**	2528.0	2322.0	2589.0	2469.0
40 × 3 × 2	**962.0**	1086.0	986.0	1019.0	1047.0	80 × 5 × 2	**1035.0**	1172.0	1043.0	1315.0	1136.0
40 × 3 × 4	**1463.0**	1684.0	1505.0	1581.0	1636.0	80 × 5 × 4	**1747.0**	1986.0	1777.0	2303.0	1876.0
40 × 3 × 6	**1608.0**	1777.0	1641.0	1728.0	1729.0	80 × 5 × 6	**1818.0**	2096.0	1856.0	2472.0	2017.0
40 × 3 × 8	**1610.0**	1775.0	1621.0	1939.0	1756.0	80 × 5 × 8	**1860.0**	2121.0	1904.0	2279.0	2002.0
40 × 4 × 2	**917.0**	1083.0	949.0	1072.0	1040.0	100 × 2 × 2	**3107.0**	3253.0	3132.0	3245.0	3202.0
40 × 4 × 4	**1160.0**	1290.0	1193.0	1289.0	1296.0	100 × 2 × 4	**4912.0**	5041.0	4944.0	5020.0	5003.0
40 × 4 × 6	**1222.0**	1392.0	1244.0	1434.0	1348.0	100 × 2 × 6	**4782.0**	5038.0	4819.0	4929.0	4932.0
40 × 4 × 8	**1374.0**	1590.0	1422.0	1701.0	1524.0	100 × 2 × 8	**4919.0**	5214.0	4954.0	5043.0	5107.0
40 × 5 × 2	**677.0**	798.0	705.0	869.0	757.0	100 × 3 × 2	**2101.0**	2320.0	2140.0	2251.0	2239.0
40 × 5 × 4	**970.0**	1179.0	1002.0	1214.0	1126.0	100 × 3 × 4	**3237.0**	3510.0	3296.0	3460.0	3415.0
40 × 5 × 6	**1160.0**	1344.0	1214.0	1419.0	1322.0	100 × 3 × 6	**3098.0**	3302.0	3114.0	3332.0	3244.0
40 × 5 × 8	**1349.0**	1537.0	1375.0	1642.0	1581.0	100 × 3 × 8	**3683.0**	3933.0	3726.0	4198.0	3925.0
60 × 2 × 2	**1699.0**	1803.0	1721.0	1843.0	1756.0	100 × 4 × 2	**1649.0**	1798.0	1696.0	1974.0	1748.0
60 × 2 × 4	**2873.0**	3076.0	2880.0	3032.0	2988.0	100 × 4 × 4	**2609.0**	2894.0	2650.0	2886.0	2753.0
60 × 2 × 6	**3072.0**	3372.0	3107.0	3221.0	3250.0	100 × 4 × 6	**2738.0**	3017.0	2778.0	3105.0	2948.0
60 × 2 × 8	**2922.0**	3079.0	2949.0	3029.0	3028.0	100 × 4 × 8	**2605.0**	2892.0	2655.0	2863.0	2803.0
60 × 3 × 2	**2039.0**	2206.0	2042.0	2244.0	2161.0	100 × 5 × 2	**1903.0**	2101.0	1921.0	2357.0	2044.0
60 × 3 × 4	**2145.0**	2378.0	2186.0	2328.0	2293.0	100 × 5 × 4	**1414.0**	1582.0	1448.0	1718.0	1583.0
60 × 3 × 6	**2255.0**	2455.0	2273.0	2832.0	2429.0	100 × 5 × 6	**1732.0**	1952.0	1779.0	2191.0	1889.0
60 × 3 × 8	**2366.0**	2545.0	2412.0	2463.0	2490.0	100 × 5 × 8	**2264.0**	2528.0	2313.0	2782.0	2415.0
120 × 2 × 2	3536.0	3549.0	3551.0	3524.0	**3519.0**	140 × 2 × 2	**4400.0**	4449.0	4488.0	4414.0	4424.0
120 × 2 × 4	**5766.0**	5770.0	5788.0	5776.0	5773.0	140 × 2 × 4	**6539.0**	6550.0	6552.0	6541.0	6551.0
120 × 2 × 6	6069.0	6083.0	6107.0	**6033.0**	6085.0	140 × 2 × 6	6771.0	6832.0	6800.0	**6697.0**	6864.0
120 × 2 × 8	**5861.0**	5883.0	5864.0	5869.0	5880.0	140 × 2 × 8	**6671.0**	6747.0	6714.0	6699.0	6710.0
120 × 3 × 2	**2359.0**	2367.0	2365.0	2393.0	2370.0	140 × 3 × 2	**2632.0**	2684.0	2698.0	2637.0	2681.0
120 × 3 × 4	2570.0	2602.0	2605.0	2632.0	**2564.0**	140 × 3 × 4	**2795.0**	2927.0	2864.0	2796.0	2852.0
120 × 3 × 6	**4061.0**	4091.0	4089.0	4344.0	4069.0	140 × 3 × 6	4805.0	4820.0	**4786.0**	5284.0	4813.0
120 × 3 × 8	**4455.0**	4506.0	4496.0	4628.0	4515.0	140 × 3 × 8	**3565.0**	3694.0	3654.0	3582.0	3640.0
120 × 4 × 2	1865.0	1907.0	1914.0	2013.0	**1862.0**	140 × 4 × 2	**2072.0**	2079.0	2086.0	2531.0	2096.0
120 × 4 × 4	**2235.0**	2331.0	2285.0	2533.0	2237.0	140 × 4 × 4	2605.0	2656.0	2619.0	2754.0	**2557.0**
120 × 4 × 6	**2420.0**	2472.0	2492.0	2507.0	2479.0	140 × 4 × 6	**3705.0**	3707.0	3779.0	4378.0	3715.0
120 × 4 × 8	3322.0	**3295.0**	3304.0	3547.0	3307.0	140 × 4 × 8	**4003.0**	4008.0	4069.0	4196.0	4066.0
120 × 5 × 2	**2472.0**	2501.0	2525.0	3110.0	2474.0	140 × 5 × 2	**2022.0**	2066.0	2055.0	2272.0	2027.0
120 × 5 × 4	2757.0	2834.0	2766.0	3245.0	**2743.0**	140 × 5 × 4	**3052.0**	3057.0	3110.0	3448.0	3102.0
120 × 5 × 6	**2719.0**	2841.0	2809.0	3184.0	2729.0	140 × 5 × 6	**3368.0**	3423.0	3431.0	3736.0	3378.0
120 × 5 × 8	**2806.0**	2825.0	2859.0	3232.0	2844.0	140 × 5 × 8	**3502.0**	3508.0	3514.0	3949.0	3532.0

Table 6. Computational results of instances of five algorithms on *AVG*.

Instance	CSFLA	SFLA	IDCOA	HVNS	IWOA	Instance	CSFLA	SFLA	IDCOA	HVNS	IWOA
20 × 2 × 2	**677.6**	758.8	702.3	725.2	738.4	60 × 4 × 2	**1051.7**	1182.7	1073.3	1150.5	1135.8
20 × 2 × 4	**1086.9**	1192.0	1113.0	1126.1	1157.8	60 × 4 × 4	**1674.2**	1859.6	1705.0	1840.2	1794.1
20 × 2 × 6	**1361.6**	1448.6	1393.0	1396.6	1430.2	60 × 4 × 6	**1265.0**	1420.1	1295.2	1391.8	1370.5
20 × 2 × 8	**1535.6**	1702.1	1564.4	1594.0	1655.6	60 × 4 × 8	**1725.5**	1890.4	1752.8	1919.3	1830.2
20 × 3 × 2	**718.9**	817.6	741.6	791.2	786.8	60 × 5 × 2	**1193.9**	1351.5	1222.1	1423.6	1281.7
20 × 3 × 4	**919.2**	1039.2	954.4	979.2	1015.8	60 × 5 × 4	**1283.0**	1480.7	1307.1	1496.2	1396.0
20 × 3 × 6	**867.2**	961.8	884.1	939.5	953.2	60 × 5 × 6	**1513.2**	1718.9	1553.3	1780.4	1651.9
20 × 3 × 8	**1173.7**	1310.1	1200.2	1219.9	1283.5	60 × 5 × 8	**1634.9**	1834.1	1674.7	1858.5	1766.4
20 × 4 × 2	**413.0**	472.7	430.3	442.7	452.2	80 × 2 × 2	**3858.1**	3936.8	3868.0	3920.1	3904.2
20 × 4 × 4	**594.4**	687.9	615.1	641.8	684.3	80 × 2 × 4	**2640.7**	2806.6	2661.9	2720.8	2762.9
20 × 4 × 6	**830.9**	942.0	857.4	895.2	921.3	80 × 2 × 6	**3746.5**	3959.1	3764.6	3845.6	3894.4
20 × 4 × 8	**933.8**	1038.5	944.2	1005.9	1009.9	80 × 2 × 8	**4218.4**	4510.8	4267.7	4343.0	4418.1
20 × 5 × 2	**470.4**	571.9	496.3	571.4	541.0	80 × 3 × 2	**1464.1**	1606.9	1496.4	1619.0	1572.1
20 × 5 × 4	**621.1**	713.6	631.6	748.4	682.6	80 × 3 × 4	**1566.7**	1712.7	1595.7	1662.2	1672.0
20 × 5 × 6	**825.1**	948.4	843.9	928.2	916.2	80 × 3 × 6	**2624.5**	2776.4	2640.5	2746.8	2715.6
20 × 5 × 8	**889.0**	997.8	906.6	981.9	965.2	80 × 3 × 8	**2780.4**	2986.7	2816.1	2862.8	2900.0
40 × 2 × 2	**1363.2**	1446.2	1380.1	1409.9	1437.8	80 × 4 × 2	**1901.0**	2014.4	1915.4	2132.3	2001.0
40 × 2 × 4	**1903.0**	2017.3	1919.7	1949.8	1985.5	80 × 4 × 4	**2013.7**	2212.1	2049.9	2260.0	3132.7
40 × 2 × 6	**1658.3**	1813.4	1674.6	1705.0	1775.6	80 × 4 × 6	**2242.2**	2476.3	2284.6	2412.5	2354.8
40 × 2 × 8	**2308.1**	2442.8	2336.2	2359.5	2413.3	80 × 4 × 8	**2254.8**	2462.0	2290.3	2480.8	2405.9
40 × 3 × 2	**938.4**	1060.2	967.2	986.8	1014.7	80 × 5 × 2	**999.7**	1142.8	1030.2	1177.4	1098.4
40 × 3 × 4	**1446.1**	1605.9	1477.9	1516.4	1554.3	80 × 5 × 4	**1724.0**	1927.6	1755.0	1955.9	1835.9
40 × 3 × 6	**1588.9**	1736.5	1605.7	1650.1	1683.9	80 × 5 × 6	**1787.3**	2015.1	1826.3	2110.1	1929.8
40 × 3 × 8	**1579.6**	1725.6	1598.5	1702.2	1683.2	80 × 5 × 8	**1845.6**	2055.2	1883.2	2084.2	1952.3
40 × 4 × 2	**903.4**	1026.1	930.5	983.1	986.3	100 × 2 × 2	**3094.1**	3212.9	3105.0	3163.8	3166.5
40 × 4 × 4	**1149.4**	1268.6	1171.3	1240.6	1236.2	100 × 2 × 4	**4898.9**	5010.3	4904.5	4978.6	4968.6
40 × 4 × 6	**1199.8**	1353.3	1232.6	1326.0	1303.3	100 × 2 × 6	**4749.8**	4960.7	4784.4	4869.0	4885.1
40 × 4 × 8	**1357.0**	1531.6	1399.9	1535.4	1470.2	100 × 2 × 8	**4895.7**	5138.5	4938.7	4971.6	5037.0
40 × 5 × 2	**661.1**	765.6	688.3	771.1	735.3	100 × 3 × 2	**2088.2**	2254.8	2118.2	2175.5	2194.6
40 × 5 × 4	**957.1**	1119.3	989.8	1108.9	1063.1	100 × 3 × 4	**3212.3**	3441.0	3254.6	3356.5	3347.7
40 × 5 × 6	**1148.0**	1315.6	1180.3	1259.1	1263.1	100 × 3 × 6	**3072.9**	3239.3	3092.9	3186.2	3178.1
40 × 5 × 8	**1322.7**	1494.5	1346.1	1455.9	1458.5	100 × 3 × 8	**3648.3**	3889.2	3697.0	3843.5	3833.0
60 × 2 × 2	**1673.9**	1784.6	1691.3	1739.3	1730.9	100 × 4 × 2	**1636.4**	1769.4	1663.1	1789.0	1724.0
60 × 2 × 4	**2829.7**	3009.9	2860.1	2934.4	2947.4	100 × 4 × 4	**2585.6**	2789.4	2619.0	2743.0	2694.6
60 × 2 × 6	**3048.1**	3272.0	3074.1	3127.7	3214.0	100 × 4 × 6	**2707.3**	2954.7	2745.1	2893.4	2864.1
60 × 2 × 8	**2894.7**	3037.8	2911.2	2965.2	2983.3	100 × 4 × 8	**2570.6**	2796.7	2607.4	2754.7	2730.0
60 × 3 × 2	**2024.4**	2139.1	2033.2	2118.5	2098.8	100 × 5 × 2	**1882.9**	2063.9	1901.8	2113.7	1969.3
60 × 3 × 4	**2127.4**	2321.2	2159.4	2237.9	2258.6	100 × 5 × 4	**1399.4**	1543.3	1427.8	1586.0	1492.3
60 × 3 × 6	**2232.9**	2389.7	2247.4	2384.2	2352.9	100 × 5 × 6	**1702.3**	1901.0	1751.8	1945.8	1825.3
60 × 3 × 8	**2351.6**	2501.4	2384.5	2430.6	2451.5	100 × 5 × 8	**2242.1**	2483.8	2287.9	2504.7	2370.5
120 × 2 × 2	**3465.1**	3526.3	3513.7	3465.3	3501.8	140 × 2 × 2	4347.9	4406.0	4403.7	**4341.4**	4379.8
120 × 2 × 4	**5718.8**	5746.6	5749.7	5730.2	5740.3	140 × 2 × 4	**6481.8**	6527.7	6527.0	6484.7	6536.4
120 × 2 × 6	6004.3	6058.6	6067.9	**5996.2**	6054.9	140 × 2 × 6	**6659.5**	6791.3	6742.4	6670.5	6763.9
120 × 2 × 8	**5774.9**	5823.3	5817.8	5778.2	5822.5	140 × 2 × 8	**6591.9**	6677.5	6665.9	6597.7	6665.3
120 × 3 × 2	2254.5	2319.6	2320.8	**2251.9**	2304.8	140 × 3 × 2	**2579.2**	2654.2	2651.9	2587.7	2641.8
120 × 3 × 4	**2497.0**	2533.4	2526.2	2537.4	2534.8	140 × 3 × 4	**2738.7**	2831.5	2805.4	2746.7	2805.9
120 × 3 × 6	**3967.7**	4044.4	4006.5	4036.4	3972.4	140 × 3 × 6	**4734.6**	4752.2	4751.4	4807.3	4745.0
120 × 3 × 8	**4378.2**	4473.7	4453.6	4435.5	4459.2	140 × 3 × 8	**3501.2**	3605.0	3613.3	3512.4	3585.5
120 × 4 × 2	**1799.1**	1852.3	1837.5	1877.7	1818.5	140 × 4 × 2	2037.9	2058.0	2060.4	2202.6	**2034.0**
120 × 4 × 4	**2203.7**	2273.2	2243.1	2327.0	2220.8	140 × 4 × 4	2512.0	2532.8	2530.0	2593.1	**2511.6**
120 × 4 × 6	**2388.6**	2428.9	2453.9	2424.8	2425.8	140 × 4 × 6	**3609.3**	3648.7	3684.9	3857.9	3658.4
120 × 4 × 8	**3233.4**	3243.3	3235.2	3327.4	3236.9	140 × 4 × 8	**3933.7**	3966.3	4004.6	3946.3	3994.4
120 × 5 × 2	**2397.5**	2443.3	2424.1	2673.7	2415.5	140 × 5 × 2	**1969.6**	1999.7	1981.8	2038.5	1977.3
120 × 5 × 4	**2635.4**	2693.5	2702.9	2928.0	2689.8	140 × 5 × 4	**2957.1**	3015.5	3056.7	3107.6	3023.7
120 × 5 × 6	**2678.0**	2706.5	2725.4	2864.0	2693.7	140 × 5 × 6	**3289.5**	3340.7	3349.8	3477.8	3304.2
120 × 5 × 8	**2739.1**	2771.3	2777.6	2919.0	2782.5	140 × 5 × 8	**3378.6**	3467.8	3429.6	3466.8	3420.0

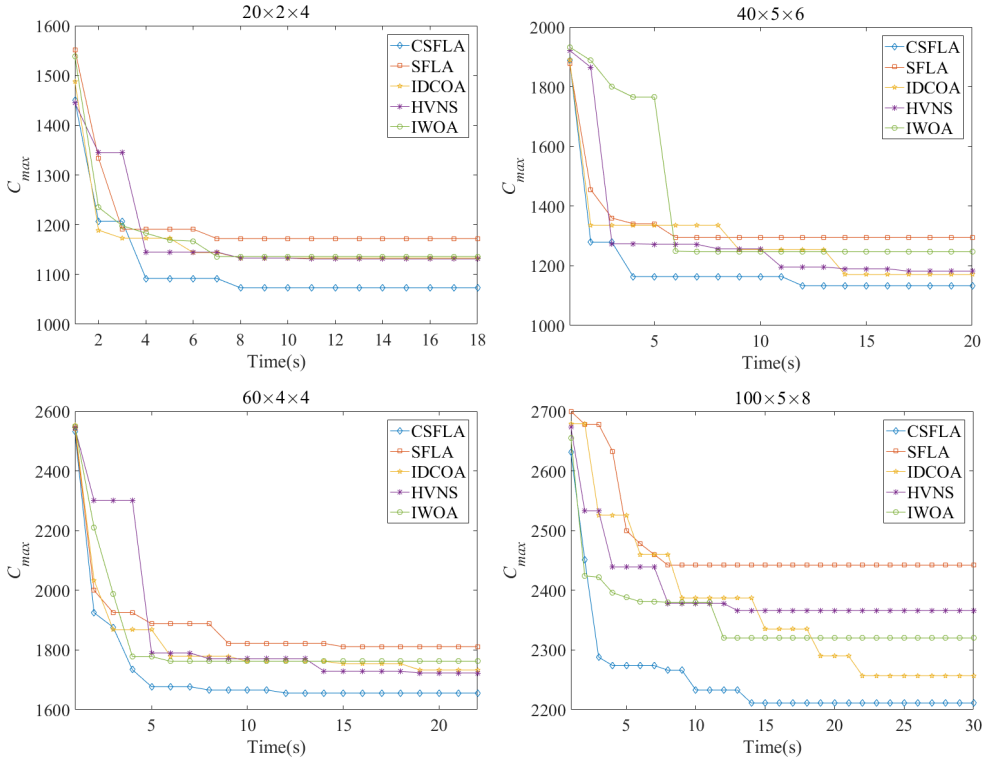

Figure 3. Convergence curves of five algorithms.

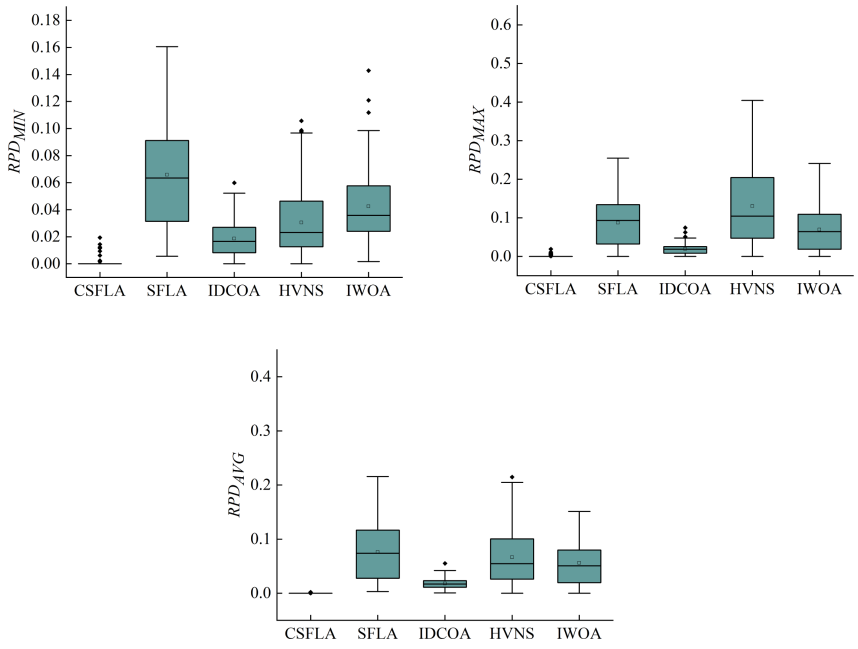

Figure 4. Box plots of five algorithms.

Table 7. Results of Wilcoxon test.

Wilcoxon-Test	*MIN*	*MAX*	*AVG*
Wilcoxon test (CSFLA, SFLA)	0.000	0.000	0.000
Wilcoxon test (CSFLA, IDCOA)	0.000	0.000	0.000
Wilcoxon test (CSFLA, HVNS)	0.000	0.000	0.000
Wilcoxon test (CSFLA, IWOA)	0.000	0.000	0.000

As shown in Table 4, CSFLA obtains smaller MIN than SFLA on all instances and MIN of CSFLA is lower than that of SFLA by at least 50 of 101 instances. CSFLA converges better than SFLA. It can be found from Table 5 that MAX of CSFLA is better than that of SFLA in 111 of 112 instances and SFLA is worse MAX than CSFLA in at least 50 of 91 instances. CSFLA possesses better stability than SFLA. Table 6 show that CSFLA obtains smaller AVG than SFLA on all instances and AVG of CSFLA is better than SFLA in at least 50 f 96 instances. CSFLA has a better average performance than SFLA. The significant performance differences between CSFLA and SFLA also can be seen from Table 7 and Figures 3 and 4. The new strategies such as the three cooperations really have a positive impact on the performance of CSFLA, so the new strategies are effective.

Table 4 show that CSFLA performs better than IDCOA, HVNS, and IWOA on MIN. CSFLA produces smaller MIN than with three comparative algorithms on 104 of 112 instances; moreover, MIN of CSFLA is less than that of IDCOA by at least 50 in 26 instances, HVNS by at least 50 in 45 instances, and IWOA by at least 50 in 88 instances. CSFLA converges better than the three comparative algorithms. The results in Table 7, Figures 3 and 4 also reveal the convergence advantage of CSFLA.

As stated in Table 5, CSFLA produces smaller MAX than three comparative algorithms in 102 instances. CSFLA obtains smaller MAX than IDCOA by at least 50 in 20 instances; MAX of CSFLA is less than that of HVNS by at least 50 in 95 instances and IWOA produces a larger MAX by at least 50 in 86 instances. CSFLA performs better than its comparative algorithms on stability performance. Figure 4 and Table 7 also depict the obvious stability performance difference between CSFLA and its comparative algorithms.

It also can be found from Table 6 that CSFLA outperforms its three comparative algorithms on AVG. CSFLA generates better AVG than its comparative algorithms in 107 instances. The advantages of CSFLA on average performance can also can be drawn from Table 7 and Figure 4.

CSFLA possesses three cooperations, two of which are used between two memeplexes of groups 1 and 2, and one of which is performed between group 1 and group 2. The adaptive search strategy is also used in search process of each group. These cooperations can make full use of the good solutions in memeplexes of group 1 and avoid the computing resource waste in group 2; as a result, exploration is intensified effectively and high diversity can be kept. The periodical population shuffling can result in good solution structure of memeplexes in T generations. The exploitation ability is also enhanced, so the new strategies of CSFLA can make a good balance between exploration and exploitation, and CSFLA produces promising results on DAHFSP with factory eligibility, which enriches the optimization algorithm mechanism to solve the analyzed production scheduling problem.

5. Conclusions and Future Topics

DAHFSP with transportation and factory eligibility is seldom considered. In this study, a new algorithm named CSFLA and based on three cooperations is presented to solve DAHFSP with factory eligibility. The problem-related feature is used. Memeplexes are evaluated, group 1 (with the two best memeplexes) and group 2 (with the two worst memeplexes) are formed. A new cooperation between memeplexes and an adaptive search strategy are implemented in each group. An adaptive cooperation between groups 1 and 2 is also given. Population shuffling is executed every T generations. Extensive computational experiments are conducted on 112 instances. Computational results demonstrate that new strategies are effective and CSFLA is a very competitive algorithm for the considered DAHFSP with makespan minimization. However, in real-life manufacturing exists complex processing environments and constraints, which may cause limitations for this research.

Production scheduling problems exist in the real-life manufacturing processes such as the casting process and engine assembly plant, and these problems have some special features. For example, batch processing machines exist in the last stage of HFSP in the steelmaking continuous-casting process. We will focus on production scheduling problems in the casting process or other real-world manufacturing issues, and try to solve them

by applying the characteristics of the problems and the new optimization mechanisms of meta-heuristics.

Author Contributions: Conceptualization, D.L.; methodology, T.D.; software, T.D.; validation, D.L.; formal analysis, D.L.; investigation, D.L.; resources, T.D.; data curation, T.D.; writing-original draft and review, D.L.; visualization, T.D.; supervision, D.L.; project administration, D.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China (No. 61573264).

Data Availability Statement: Not applicable.

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

Abbreviations

SDST	sequence-dependent setup time
$gbest$	the best solution of population
st_{jilf}	the start time of the fabfication of com_{ij} at stage l in factory f
et_{jilf}	the end time of the fabfication of com_{ij} at stage l in factory f
tst_{if}	the start time of transportation of producth i in factory f
tet_{if}	the end time of transportation of producth i in factory f
ast_{if}	the start time of assembly of product i in factory f
aet_{if}	the end time of assembly of product i in factory f
U	a large positive number
X_{jif}	decision variable, if com_{ij} is allocated in factory f, $X_{jif} = 1$; otherwise $X_{jif} = 0$
Y_{jiflk}	decision variable, if com_{ij} is allocated in M_{flk}, $Y_{jiflk} = 1$; otherwise $Y_{jiflk} = 0$
$Z_{jj'ifl}$	decision variable, if com_{ij} is processed before $com_{ij'}$ at stage l in factory f, $Z_{jj'ifl} = 1$; otherwise $Z_{jj'ifl} = 0$
A_{if}	decision variable, if product i is transported by TM_f, $A_{if} = 1$; otherwise $A_{if} = 0$
$B_{ii'f}$	decision variable, if product i is transported before product i' in factory f, $B_{ii'f} = 1$; otherwise $B_{ii'f} = 0$
$D_{ii'f}$	decision variable, if product i is assembled before product i' in factory f, $D_{ii'f} = 1$; otherwise $D_{ii'f} = 0$
E_{if}	decision variable, if product i is assembled by AM_f, $E_{if} = 1$; otherwise $E_{if} = 0$

References

1. Hao, J.H.; Li, J.Q.; Du, Y.; Song, M.X.; Duan, P.; Zhang, Y.Y. Solving distributed hybrid flowshop scheduling problems by a hybrid brain storm optimization algorithm. *IEEE Access* **2019**, *7*, 68879–68894. [CrossRef]
2. Zheng, J.; Wang, L.; Wang, J.J. A cooperative coevolution algorithm for multi-objective fuzzy distributed hybrid flow shop. *Knowl. Based Syst.* **2020**, *194*, 105536. [CrossRef]
3. Shao, W.S.; Shao, Z.S.; Pi, D.C. Modeling and multi-neighborhood iterated greedy algorithm for distributed hybrid flow shop scheduling problem. *Knowl. Based Syst.* **2020**, *194*, 105527. [CrossRef]
4. Shao, W.S.; Shao, Z.S.; Pi, D.C. Effective constructive heuristics for distributed no-wait flexible flow shop scheduling problem. *Comput. Oper. Res.* **2021**, *136*, 105482. [CrossRef]
5. Shao, W.S.; Shao, Z.S.; Pi, D.C. Multi-objective evolutionary algorithm based on multiple neighborhoods local search for multi-objective distributed hybrid flow shop scheduling problem. *Expert Syst. Appl.* **2021**, *183*, 115453. [CrossRef]
6. Li, Y.L.; Li, X.Y.; Gao, L.; Zhang, B.; Pan, Q.K.; Tasgetiren, F.; Meng, L.L. A discrete artificial bee colony algorithm for distributed hybrid flowshop scheduling problem with sequence-dependent setup times. *Int. J. Prod. Res.* **2021**, *59*, 3880–3899. [CrossRef]
7. Wang, J.J.; Wang, L. A cooperative memetic algorithm with learning-based agent for energy-aware distributed hybrid flow-shop scheduling. *IEEE Trans. Evol. Comput.* **2022**, *26*, 461–475. [CrossRef]
8. Geng, K.F.; Ye, C.M. A memetic algorithm for energy-efficient distributed re-entrant hybrid flow shop scheduling problem. *J. Intell. Fuzzy Syst.* **2021**, *41*, 3951–3971. [CrossRef]
9. Qin, H.; Li, T.; Teng, Y.; Wang, K. Integrated production and distribution scheduling in distributed hybrid flow shops. *Memet. Comput.* **2021**, *13*, 185–202. [CrossRef]

10. Shao, Z.S.; Shao, W.S.; Pi, D.C. A learning-Based Selection Hyper-Heuristic for Distributed Heterogeneous Hybrid Blocking Flow-shop Scheduling. *IEEE Trans. Emerg. Top. Comput. Intell.* **2022**. Available online: https://ieeexplore.ieee.org/abstract/document/9782097 (accessed on 22 February 2023). [CrossRef]
11. Qin, H.X.; Han, Y.Y. A collaborative iterative greedy algorithm for the scheduling of distributed heterogeneous hybrid flow shop with blocking constraints. *Expert Syst. Appl.* **2022**, *201*, 117256. [CrossRef]
12. Cai, J.C.; Lei, D.M.; Wang, J.; Wang, L. A novel shuffled frog-leaping algorithm with reinforcement learning for distributed assembly hybrid flow shop scheduling. *Int. J. Prod. Res.* **2022**. Available online: https://www.tandfonline.com/doi/abs/10.1080/00207543.2022.2031331 (accessed on 22 February 2023). [CrossRef]
13. Ying, K.C.; Lin, S.W. Minimizing makespan for the distributed hybrid flowshop scheduling problem with multiprocessor tasks. *Expert Syst. Appl.* **2018**, *92*, 132–141. [CrossRef]
14. Cai, J.C.; Zhou, R.; Lei, D.M. Dynamic shuffled frog-leaping algorithm for distributed hybrid flow shop scheduling with multiprocessor tasks. *Eng. Appl. Artif. Intell.* **2020**, *90*, 103540. [CrossRef]
15. Jiang, E.D.; Wang, L.; Wang, J.J. Decomposition-based multi-objective optimization for energy-aware distributed hybrid flow shop scheduling with multiprocessor tasks. *Tsinghua Sci. Technol.* **2021**, *26*, 646–663. [CrossRef]
16. Li, Y.L.; Li, X.Y.; Gao, L.; Meng, L.L. An improved artificial bee colony algorithm for distributed heterogeneous hybrid flow-shop scheduling problem with sequence-dependent setup times. *Comput. Ind. Eng.* **2020**, *147*, 106638. [CrossRef]
17. Cai, J.C.; Lei, D.M.; Li, M. A shuffled frog-leaping algorithm with memeplex quality for bi-objective distributed scheduling in hybrid flow shop. *Int. J. Prod. Res.* **2021**, *59*, 5404–5421. [CrossRef]
18. Lei, D.M.; Wang, T. Solving distributed two-stage hybrid flow-shop scheduling using a shuffled frog-leaping algorithm with memeplex grouping. *Eng. Optimiz.* **2020**, *52*, 1461–1474. [CrossRef]
19. Lei, D.M.; Xi, B.J. Diversified teaching-learning-based optimization for fuzzy two-stage hybrid flow shop scheduling with setup time. *J. Intell. Fuzzy Syst.* **2021**, *41*, 4159–4173. [CrossRef]
20. Hatami, S.; Ruiz, R.; Carlos, A.R. The distributed assembly permutation flowshop scheduling problem. *Int. J. Prod. Res.* **2013**, *51*, 5292–5308. [CrossRef]
21. Xiong, F.L.; Xing, K.Y.; Wang, F.; Lei, H.; Han, L.B. Minimizing the total completion time in a distributed two stage assembly system with setup times. *Comput. Oper. Res.* **2014**, *47*, 92–105. [CrossRef]
22. Zhang, G.H.; Xing, K.Y. Memetic social spider optimization algorithm for scheduling two-stage assembly flowshop in a distributed environment. *Comput. Ind. Eng.* **2018**, *125*, 423–433. [CrossRef]
23. Zhang, Z.Q.; Qian, B.; Hu, R.; Jin, H.P.; Wang, L. A matrix-cube-based estimation of distribution algorithm for the distributed assembly permutation flow-shop scheduling problem. *Swarm Evol. Comput.* **2021**, *60*, 100785. [CrossRef]
24. Wang, J.; Lei, D.M.; Cai, J.C. An adaptive artificial bee colony with reinforcement learning for distributed three-stage assembly scheduling with maintenance. *Appl. Soft. Comput.* **2022**, *117*, 108371. [CrossRef]
25. Huang, J.L.; Gu, X.S. Distributed assembly permutation flow-shop scheduling problem with sequence-dependent set-up times using a novel biogeography-based optimization algorithm. *Eng. Optimiz.* **2022**, *54*, 593–613. [CrossRef]
26. Eusuff, M.; Lansey, K.; Pasha, F. Shuffled frog-leaping algorithm: A memetic meta-heuristic for discrete optimization. *Eng. Optimiz.* **2006**, *38*, 129–154. [CrossRef]
27. Cai, J.C.; Lei, D.M. A cooperated shuffled frog-leaping algorithm for distributed energy-efficient hybrid flow shop scheduling with fuzzy processing time. *Complex Intell. Syst.* **2021**, *7*, 2235–2253. [CrossRef]
28. Rahimi-Vahed, A.; Mirzaei, A.H. Solving a bi-criteria permutation flow-shop problem using shuffled frog-leaping algorithm. *Soft Comput.* **2008**, *12*, 435–452. [CrossRef]
29. Pan, Q.K.; Wang, L.; Gao, L.; Li, J.Q. An effective shuffled frog-leaping algorithm for lot-streaming flow shop scheduling problem. *Int. J. Adv. Manuf. Technol.* **2011**, *52*, 699–713. [CrossRef]
30. Xu, Y.; Wang, L.; Wang, S.Y.; Liu, M. An effective shuffled frog-leaping algorithm for solving the hybrid flow-shop scheduling problem with identical parallel machines. *Eng. Optimiz.* **2013**, *45*, 1409–1430. [CrossRef]
31. Wang, L.; Li, D.D. Fuzzy distributed hybrid flow shop scheduling problem with heterogeneous factory and unrelated parallel machine: A shuffled frog leaping algorithm with collaboration of multiple search strategies. *IEEE Access.* **2020**, *8*, 191191–191203. [CrossRef]
32. Li, X.; Luo, J.P.; Chen, M.R.; Wang, N. An improved shuffled frog-leaping algorithm with extremal optimisation for continuous optimisation. *Inf. Sci.* **2012**, *192*, 143–151. [CrossRef]
33. Zhang, X.X.; Ji, Z.C.; Wang, Y. An improved SFLA for flexible job shop scheduling problem considering energy consumption. *Mod. Phys. Lett. B* **2018**, *32*, 1840112. [CrossRef]
34. Zhang, H.; Liu, S.; Moraca, S.; Ojstersek, R. An effective use of hybrid metaheuristics algorithm for job shop scheduling problem. *Int. J. Simul. Model.* **2018**, *16*, 644–657. [CrossRef]
35. Kong, M.; Liu, X.B.; Pei, J.; Pardalos, P.M.; Mladenovic, N. Parallel-batching scheduling with nonlinear processing times on a single and unrelated parallel machines. *J. Glob. Optim.* **2020**, *78*, 693–715. [CrossRef]
36. Xu, Y.; Wang, L.; Liu, M.; Wang, S.Y. An effective shuffled frog-leaping algorithm for hybrid flow-shop scheduling with multiprocessor tasks. *Int. J. Adv. Manuf. Technol.* **2013**, *68*, 1529–1537. [CrossRef]
37. Zhang, G.H.; Xing, K.Y.; Cao, F. Scheduling distributed flowshops with flexible assembly and set-up time to minimise makespan. *Int. J. Prod. Res.* **2018**, *56*, 3226–3244. [CrossRef]

38. Komaki, G.M.; Teymourian, E.; Kayvanfar, V.; Booyavi, Z. Improved discrete cuckoo optimization algorithm for the three-stage assembly flowshop scheduling problem. *Comput. Ind. Eng.* **2017**, *105*, 158–173. [CrossRef]
39. Li, Q.H.; Li, J.Q.; Zhang, Q.K.; Duan, P.; Meng, T. An improved whale optimisation algorithm for distributed assembly flow shop with crane transportation. *Int. J. Autom. Control.* **2021**, *15*, 710–743. [CrossRef]
40. Montgomery, D.C. *Design and Analysis of Experiments*; Wiley: Hoboken, NJ, USA, 2008.