processes

MDPI

Article

Optimizing Production Schedules: Balancing Worker Cooperation and Learning Dynamics in Seru Systems

Weiguo Liu *, Weizhe Dai and Xuyin Wang

Business School, Northwest Normal University, Lanzhou 730070, China; daiwz001025@163.com (W.D.); humorgirlwxy@nwnu.edu.cn (X.W.)
* Correspondence: winterplum.home@nwnu.edu.cn

Abstract: This paper aims to investigate the seru scheduling problem while considering the dual effects of worker cooperation and learning behavior to minimize the makespan and order processing time. Given the complexity of this research problem, an improved shuffled frog leaping algorithm based on a genetic algorithm is proposed. We design a double-layer encoding based on the problem, introduce a single point and uniform crossover operator, and select the crossover method in probability form to complete the evolution of the meme group. To avoid damaging grouping information, the individual encoding structure is transformed into unit form. Finally, numerical experiments were conducted using numerical examples of large and small sizes for verification. The experimental results demonstrate the feasibility of the proposed model and algorithm, as well as the necessity of considering worker dual behavior in the seru scheduling problem.

Keywords: seru scheduling; worker cooperation; learning effects; shuffled frog leaping algorithm

check for updates

Citation: Liu, W.; Dai, W.; Wang, X. Optimizing Production Schedules: Balancing Worker Cooperation and Learning Dynamics in Seru Systems. *Processes* **2024**, *12*, 38. https://doi.org/10.3390/pr12010038

Academic Editor: Michael C. Georgiadis

Received: 6 December 2023
Revised: 17 December 2023
Accepted: 19 December 2023
Published: 22 December 2023

1. Introduction

To efficiently and flexibly respond to the market and meet the diverse needs of consumers, the seru system has emerged. The seru system is an outcome produced by transforming the traditional assembly line disassembly, efficiently combining efficiency and flexibility [1,2]. It is known as the "ecological and economic" manufacturing model in the Japanese manufacturing industry [3]. After the application of seru systems by manufacturing companies such as Canon, Sony, and Panasonic, it has been found that seru systems also possess advantages such as reduced production costs, decreased production time, and minimal energy consumption [4–8]. Therefore, the seru system is deemed one of the most promising production methods in the era of Industry 4.0.

The current research on seru systems mainly focuses on seru construction and seru scheduling. Seru scheduling relies on the seru construction process to determine the order allocation and processing order of each seru unit. Additionally, seru scheduling is the key to reflecting the efficiency and flexibility of seru systems, and whether the advantages of seru systems can be fully utilized mainly depends on the core step of seru scheduling. In terms of seru scheduling, Zhan et al. [9] proposed the GP-SS scheduling rule to address the seru scheduling problem with resource conflicts. Jiang et al. [10] transformed the seru scheduling problem into an assignment problem. Li et al. [11] studied the on-line seru scheduling problem while considering resource conflicts. Wu et al. [12] devised a reinforcement learning-driven two-stage evolutionary algorithm to address the scheduling problem of a hybrid seru system considering worker transfer. Lian et al. [13] solved the energy-saving scheduling problem by considering seru reconfiguration. Zhang et al. [14] used the branch and bound algorithm to solve the seru scheduling problem. Shen et al. [15] designed a hybrid GA-PSO algorithm to solve the seru scheduling problem that considered worker learning effects and dynamic resource allocation. Zhang et al. [16–18] constructed scenarios for different seru scheduling problems that considered worker learning behavior and demonstrated that considering worker learning behavior can help improve seru system

performance. Jiang et al. [19] found that as the learning effect of workers increases, the production cost of the seru system will decrease. Based on the aforementioned research findings, the seru scheduling problem has been extensively explored from various angles and has achieved significant progress. However, due to the difficulty in measuring worker cooperation and learning behavior, few research studies on seru scheduling simultaneously considered worker dual behavior. The seru system is centered around workers, and the two most prominent behaviors are worker cooperation and learning behavior. Its production efficiency and performance largely depend on the workers' abilities and personal behavior. Therefore, how to effectively leverage the dual behavioral effects of workers to optimize the performance of the seru system is a crucial issue that needs to be addressed urgently.

In the production workshop, when workers repeatedly operate the same process or task, there will be a learning effect [20–22]. The learning effect refers to the phenomenon where the processing time of a single product decreases with the increase in cumulative production volume [23]. At the theoretical level, Wright [24] proposed the learning effect curve for the first time in the aviation manufacturing industry. Subsequently, learning effect models, such as the S-curve [25], Stanford-B [26], Dejong [27], and Plateau [28], were proposed based on different research scenarios. At the application level, Zhang et al. [29] considered learning effects in constructing problem scenarios in hybrid flow shop scheduling. Hu et al. [30] constructed a single-process workshop scenario that considers learning effects. Wang et al. [31] studied the joint decision-making problem of unit manufacturing systems considering learning and forgetting effects. Learning behavior is a common phenomenon in production and manufacturing, and constructing scenarios that consider learning behavior can shorten the gap between theory and practice, which is beneficial for enterprises to make low-cost and efficient production decisions. Due to human social needs, cooperative behavior may occur among workers. Good cooperative behavior can improve the work efficiency of workers; on the contrary, work efficiency will decrease [5,32,33]. In the current study, Sakamaki [32], Cao et al. [34], and Wang et al. [33] all proved that considering worker cooperation behavior in production scheduling can help improve production efficiency. It can be seen that worker behavior is a realistic influencing factor that cannot be ignored in production scheduling.

When solving the seru scheduling problem, scholars mainly relied on improved genetic algorithms [35–39] to solve problems and have achieved certain results, but the optimization algorithm has the limitation of the solving scale and the shortage of solving quality. Therefore, this paper intends to utilize the shuffled frog leaping algorithm, which has a simple idea, fewer experimental parameters, and strong global optimization ability, to solve the problem. The shuffled frog leaping algorithm combines the advantages of the Memetic algorithm and particle swarm optimization algorithm, with a strong global search ability. Some scholars have applied it to the field of production scheduling. Moreover, the hybrid leapfrog algorithm has not yet been applied in the research of seru production systems, so it is necessary to explore the application of the hybrid leapfrog algorithm to solve seru systems, which is a further expansion of the research field of seru production systems. Drawing from the preceding analysis, this paper will explore the seru scheduling problem while considering the dual behavior of workers. The objective is to minimize both the makespan and order processing time. The improved hybrid leapfrog algorithm serves as a breakthrough. The ultimate goal is to enhance the efficiency of the seru system, propose reasonable and effective worker and order-allocation plans, and provide managers with a theoretical basis and technical support for making informed scheduling decisions.

2. Problem Description

This article adopts a segmented seru as the basic unit, which is the corresponding process assembly assigned by workers within the unit. Within a seru unit, the assembly of a product is jointly completed by multiple workers. This article focuses on three subproblems: seru-worker, seru-order, and order-worker allocation. Assuming a seru system that produces C product types, W versatile workers are allocated to N seru units for I

order production. Producing each product requires k processes, each corresponding to a skill. Because workers are multi-skilled, there are multiple matching relationships between workers and processes. In the seru unit, a worker can be responsible for multiple processes of the product, which can be discontinuous, but the corresponding processing skills for the process need to be mastered by the worker.

Figure 1 describes the problem studied in this paper. In the figure, there are six processes, three orders, two products, and five workers to build two seru units. Taking seru1 as an example, if the distribution order is i_1 and i_3, and the product c_1 is produced, the corresponding operation set is $u_1 = (k_1, k_2, k_4, k_5)$, of which worker w_1 is responsible for operations k_1 and k_2, and worker w_5 is responsible for operations k_4 and k_5.

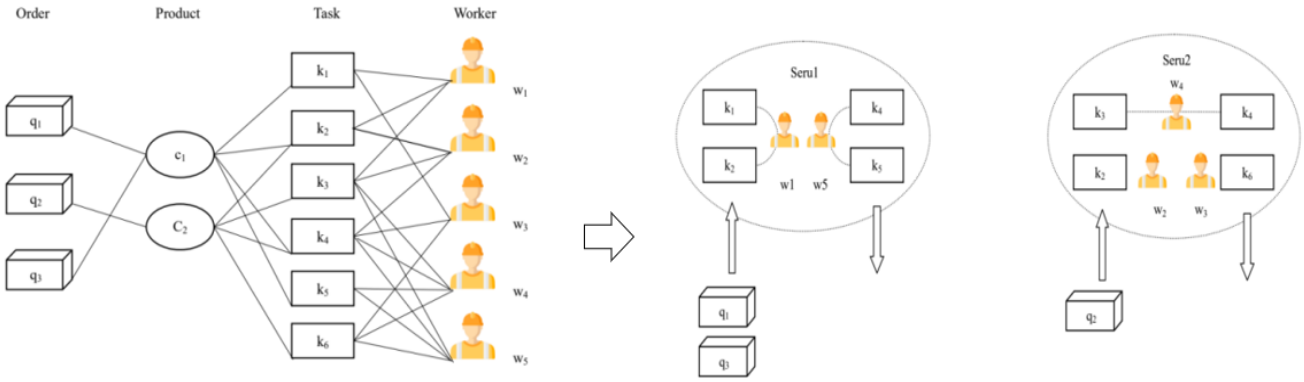

Figure 1. Problem description under seru production system.

3. Model Formulation

3.1. Model Assumptions

The research questions in this article include the following assumptions:

(1) The type and quantity of products in the order are known.
(2) Batch splitting is not considered, which means each order can only be produced within one seru unit.
(3) Each order encompasses only one type of product.
(4) The processing time of each task in each product is known and constant.
(5) The number of workers is constant.
(6) The worker movements between serus do not exist.

3.2. Modeling

This paper takes the model proposed by Wang et al. [33] as a reference, considering the influencing parameters of worker learning behavior effects, and modifies the model based on segmented units. According to the DeJong learning effect model and the characteristics of segmented units, it is necessary to calculate the cumulative number of products completed by each order worker before starting processing, as well as the cumulative number of products completed after this order is completed. This can be recorded as the upper and lower limits of the q variable cumulative completed products in the learning effect. The specific calculation formula is shown in Equations (1) and (2).

$$Q_{ijw}^{ub} = \begin{cases} Q_{jw} & , k = 1 \\ \sum_{r=1}^{j} \sum_{k=1}^{K} X_{irk} Q_{rw} C_{rp} \, C_{jp} & , k \neq 1 \end{cases} \tag{1}$$

$$Q_{ijw}^{lb} = \begin{cases} 1 & , k = 1 \\ \sum_{r=1}^{j-1} \sum_{k=2}^{K} Q_{rw} X_{ir(k-1)} X_{ijk} C_{rp} \, C_{jp} + 1 & , k \neq 1 \end{cases} \tag{2}$$

In the segmented seru unit, the cooperative behavior between workers is mainly the cooperation between workers and their left and right workers. Due to the coefficient of

cooperative behavior, a_{wu} is inversely proportional to the processing time; that is, the higher the cooperation coefficient, the shorter the processing time. Based on this, $1 - a_{wu}$ can directly represent the impact on processing time. For the overall cooperation coefficient of workers, see Equation (3).

$$a_w = (1 - a_{wu})(1 - a_{pw})Y_{wi}Y_{ui}Y_{pi} \tag{3}$$

The quantity of products completed by worker w is shown in Equation (4)

$$Q_{jw} = \sum_{i=1}^{I} Y_{wi}R_{ij}S_j \tag{4}$$

Considering the factors influencing worker cooperation and learning behavior, the total processing time for worker w to complete process s of order j is shown in Equation (5).

$$T_{jw}^{s} = \sum_{q=Q_{ijw}^{lb}}^{Q_{ijw}^{ub}} T_p^s a_w \left(M + (1 - M)q^b\right) \tag{5}$$

The model in this article aims to minimize the maximum completion time and minimize the order processing time. The specific model construction is as follows:

$$f_1 = min\ makespan = min\left\{max_{\{j\in\{1,2,\ldots J\}\}}\left(FCB_j + ST_j + FC_j\right)\right\} \tag{6}$$

$$f_2 = min\sum_{j=1}^{J}\sum_{w=1}^{W}\left(\sum_{i=1}^{I}\sum_{k=1}^{K} FC_j Y_{wi}X_{ijk}\right) \tag{7}$$

s.t.

$$\sum_{i=1}^{I} Y_{wi} = 1\ ,\mathrm{w} = 1, 2, \ldots, \mathrm{W} \tag{8}$$

$$W_{min} \le \sum_{w=1}^{W} Y_{wi} \le W_{max}\ , i = 1, 2, ..., I \tag{9}$$

$$\sum_{i=1}^{I} R_{ij} = 1, j = 1, 2, \ldots, J \tag{10}$$

$$FC_j = \frac{\sum_{i=1}^{I} X_{ijk}\sum_{w=1}^{W} Y_{wi}\sum_{p=1}^{P} U_{wp}C_{jp}\sum_{s=1}^{S} T_{jw}^{s}}{\sum_{i=1}^{I}\sum_{w=1}^{W} Y_{wi}R_{ij}} \tag{11}$$

$$ST_j = \sum_{r=1}^{J}\sum_{p=1}^{P} X_{ir(k-1)}Z_pC_{jp}\left(1 - C_{rp}\right)\ , if\ k \ne 1, r \ne j, X_{ijk} = 1 \tag{12}$$

$$FCB_j = \sum_{r=1}^{j-1}\sum_{i=1}^{I}\sum_{k=1}^{j}\left(FC_r + ST_r\right)X_{ijk}X_{ir(k-1)} \tag{13}$$

$$\sum_{j=1}^{J}\sum_{k=1}^{K} X_{ijk}FC_j < T_i\ , i = 1, 2, ..., I \tag{14}$$

$$C_j = FC_j + ST_j + FCB_j < D_j \tag{15}$$

$$X_{ijk}\ , Y_{wi} \in \{0, 1\}$$

$$S_j \in Z\ , \forall i \tag{16}$$

$$-1 \le b \le 0\ , 0 \le M \le 1$$

where Equations (6) and (7) are objective functions that represent the minimizing makespan and order processing time, respectively. Equation (8) ensures that each worker can only work within one seru unit. Equation (9) represents the number of workers within each seru unit. Equation (10) indicates that each order can only be assigned to one seru unit without considering order splitting. Equation (11) represents the processing time of all products in seru i for order j. Equation (12) represents the preparation time of order j in seru i. Equation (13) represents the start time of order j in seru i. Equation (14) limits the processing time of all orders in seru i to be less than the available production time of seru i. Equation (15) indicates that order j must be completed within the delivery date. Equation (16) represents other logical constraints in this model.

4. Improved Shuffled Frog Leaping Algorithm (SFLA)

The SFLA is a meta-heuristic algorithm that mimics the communication and cooperation among frogs during their foraging process [40]. It mainly includes four basic operations: population initialization, meme group partitioning, meme group evolution, and meme group reconstruction [41,42]. This paper proposes refinements to the hybrid leapfrog algorithm based on the characteristics of the research problem. The main contribution or improvement ideas are as follows: firstly, due to the multi-objective optimization problem in this study, there are Pareto optimality and non-dominated frontier solution sets, and the method of using a single objective function to evaluate the fitness of individuals is no longer applicable. To avoid subjective factors, this article introduces non-dominated sorting and crowding distance in the NSGA-II algorithm to evaluate individual strengths and weaknesses. Secondly, using non-dominated level grouping as the grouping result of the meme group, in order to improve the convergence speed of the algorithm, the optimal individuals from non-dominated levels are selected to form the optimal group. During the evolution process of the meme group, cross-operations are performed within the optimal group, which can better transmit optimal information. Thirdly, we introduce the single point crossover and uniform crossover operators in genetic algorithms, comprehensively utilize the advantages of the two operators, and randomly select crossover operations in the form of probability to increase population diversity.

4.1. Encoding Method

When encoding the order-scheduling problem in the seru production system, it is necessary to consider three sub-problems: worker-seru, order-seru, and order-worker. In the first two sub-problems, both workers and orders are the result of allocation in the seru unit. Due to the common feature of the seru unit, this paper proposes a double-layer gene-coding approach. The first layer of coding consists of $W + I$ gene loci, including the allocation results of two sub-problems: the first part consists of W gene loci, representing the allocation results of worker-seru; the second part consists of I gene loci, representing the allocation results of order-seru. The numbers on the first gene locus are all seru numbers. The second layer of coding represents the result of order-worker allocation. As the result of order-worker allocation is influenced by the results of the first two sub-problems, each order is considered as a unit, and the number of operations in the order is the number of gene bits under that order. Regarding the gene bits, the corresponding processing workers for that operation are represented. For example, the coding example in Figure 2 is a seru system consisting of 4 seru units, 12 workers, and 8 orders. In the first layer, the first 12 loci represent worker seru allocation, while the last 8 loci represent order seru allocation; the second layer is the worker corresponding to the process in the order. In the seru 1 unit, workers w_2, w_9, and w_{12} work together to complete production orders j_1 and j_5. In order 1, w_2 is responsible for processing J_{11} and J_{15}, w_9 is responsible for processing J_{12} and J_{14}, w_{12} is responsible for processing J_{13}, and other orders are assigned similarly.

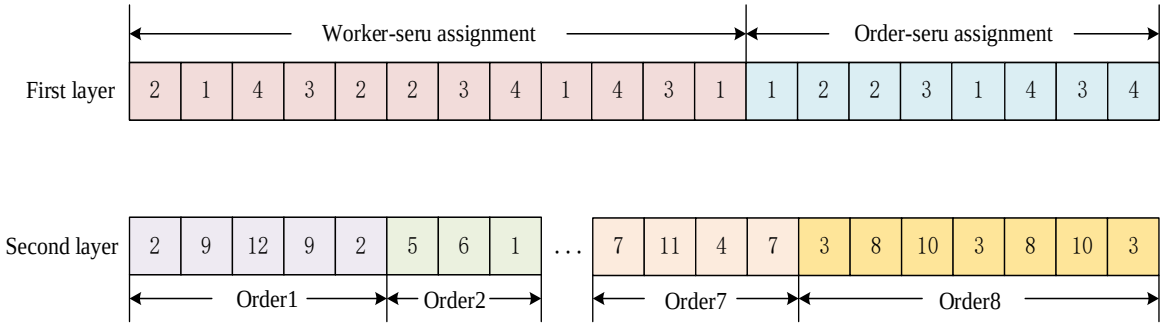

Figure 2. Encoding example diagram.

4.2. Population Initialization

Generate an initial population F consisting of N frogs in the feasible domain. Ensure that all solutions corresponding to frogs meet the coding requirements and are feasible solutions for the optimization problems. Specifically, for randomly generated frog individuals, the first layer of genes corresponds to the two parts of seru-worker and seru-order allocation. Adjust the random allocation results according to the constraints in the model. The second layer of gene coding is determined by the first layer of gene coding. The assigned order and corresponding worker must be in the same seru unit, and the worker must have the ability to process the process. Therefore, it is feasible to randomly produce initialized frog population individuals.

4.3. Division of Meme Groups

Divide the frog population into different meme groups based on non-dominated sorting and crowding distance at each level. Non-dominated sorting compares the corresponding function values of two individuals. If both objective function values of Frog 1 are better than Frog 2, it is considered that Frog 1 dominates Frog 2; if Frog 1 has an objective function value that is better than Frog 2 and another equal objective value, it is considered that Frog 1 dominates Frog 2. Count the number of non-dominated levels, denoted as Y_n. Memetic group division refers to the grouping of non-dominated levels.

4.4. Memetic Evolution

For better meme evolution, select the optimal individuals from different meme groups to form a group so that frog individuals can quickly transmit optimal information. This article uses a crossover operator to update individuals within the meme group. The crossover operation only applies to the first layer of genes, while the second layer of genes is determined based on the first layer of genes. In the improved shuffled frog leaping algorithm, each meme group has one frog with the best position and the worst position, which is the best position among all frogs. Here, the best frog individual in the group is the same as the best frog individual in the population.

During the first evolution of the meme group, a crossover operation is performed on F_b and F_w to generate a new solution. If the generated new solution is better than F_w, it is replaced. If the effect is not good, a secondary adjustment is made. The second adjustment is to randomly generate a new individual F_g to replace it. When making alternative comparisons, the fitness used in this article is the solution corresponding to the objective function one, as the solution corresponding to the objective function one is the maximum completion time of the seru production system, which better indicates the rationality of the scheduling plan.

Before performing the crossover operation, use Figure 3 to change the first-layer structure of the individual and transform it into a unit-combination form. The reason for the transformation form is, firstly, it can avoid cross operation from damaging grouping information; secondly, during the crossover process, this approach can better preserve some excellent genes of the parents; thirdly, the individual gene loci in the first layer are too long. After transforming the form, each seru unit is treated as a whole for cross-operation, which shortens the length of individual genes in the form and can improve the algorithm's solving speed.

For the seru-worker section, this article adopts a single-point crossover operation, which is simple and easy to implement and can compensate for the slow speed of uniform crossover optimization. On two parent genes, each seru unit is treated as a single gene point, and gene points are randomly selected for crossover. The crossover operation involves all genes after that gene point. In the newly generated offspring genes, since the evolution of the meme group mainly involves updating the worst frog, during the single-point crossover operation, the right gene of the worst frog was replaced with the right gene of the best frog. In order to retain some of the genes in the optimal solution, the left gene in the duplicate

gene was deleted. For missing gene loci, compensate for the missing genes in the worker coding order. The single-point crossover operation is shown in Figure 4.

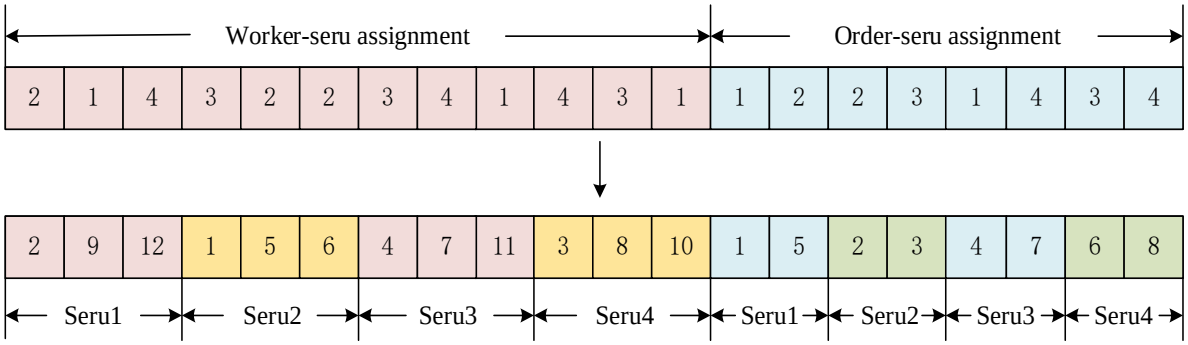

Figure 3. Individual first-layer gene transformation.

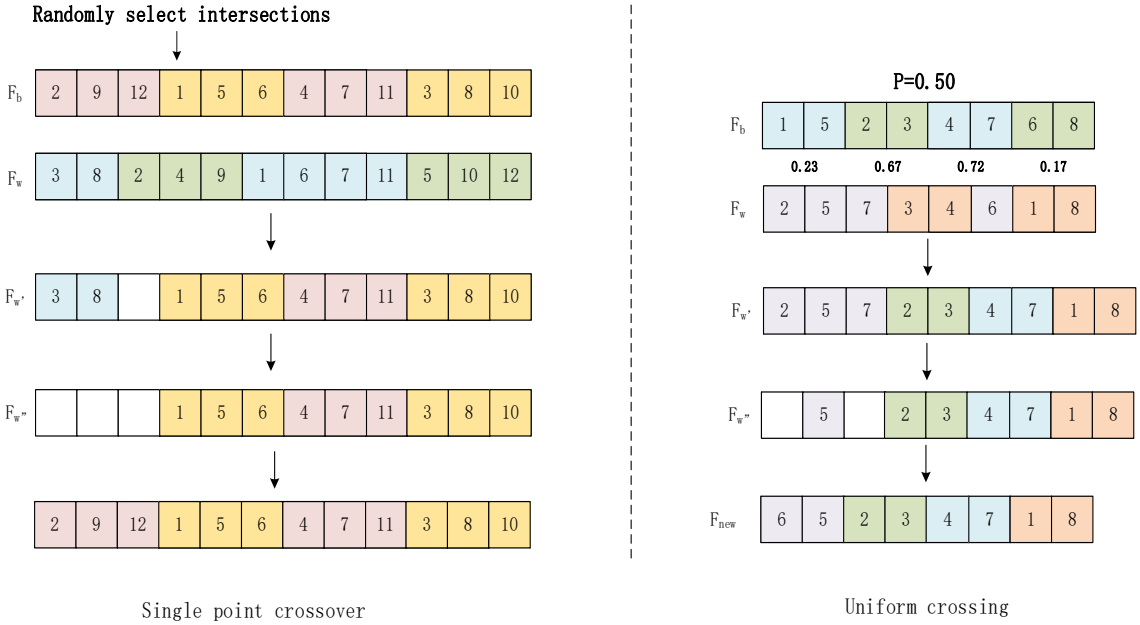

Figure 4. Cross-operation.

For the seru-order section, this article adopts the uniform crossover operation, which can compensate for the situation where a single point cross is prone to falling into local optima and increasing the diversity of the population. On two parent genes, treat each seru unit as a single gene point as a whole. For each gene point of the optimal frog individual in the parent generation, U_k is randomly generated to represent its selection probability, where $U_k \in (0, 1)$. The gene points on the parent gene are exchanged with a probability of $P = 0.5$. When the probability of corresponding gene points on the parent gene is $U_k > P$, a gene-crossover operation is performed; otherwise, no crossover operation is performed. In the process of uniform crossover, the probability of gene points is compared in order from left to right, and the operations for repeated and missing values are the same as those in the single-point crossover. The uniform crossover operation is shown in Figure 4.

To comprehensively utilize the advantages of the two operators and randomly select crossover operations in the form of probability, the single-point crossover operation of the worker-seru part and the uniform crossover operation of the order-seru part are both carried out with a 50% probability; that is, if the random selection probability $P > 50\%$, the seru-worker part of the crossover operation will be carried out, and the seru-order part will be adjusted accordingly; if $P \leq 50\%$, proceed with the cross-operation of the seru-order

section. The portion of the second gene worker order is adjusted according to changes in the first gene. The specific steps are as follows:

Step 1: Determine the optimal frog F_b and worst frog F_w in the meme group.

Step 2: Take probability P and perform single-point crossing ($P > 50\%$) or uniform crossing operation ($P \leq 50\%$) on F_b and F_w to obtain F_{new1}.

Step 3: Compare the fitness values of F_{new1} and F_w. If the fitness of F_{new1} is better than that of F_w, replace F_w with F_{new1}; otherwise, proceed to step 4.

Step 4: Randomly select another frog in the group to replace F_w.

Step 5: After updating the position of the worst frog, upgrade the cultural genome and re-rank the fitness. If the maximum number of evolutions has not been reached, return to step 1 to continue the local search until the maximum number of evolutions in the meme group is reached, ending the local search.

4.5. Algorithm Flow

The improved shuffled frog leaping algorithm proposed in this article uses non-dominated sorting and crowding distance in NSGA-II to evaluate the superiority and inferiority of individuals. To improve the convergence speed of the algorithm, the optimal individuals from each non-dominated level are selected to form a group and complete the evolution of the meme group. In the process of meme evolution, using crossover operators to update frog individuals can improve population diversity. The combination of single-point crossover and uniform crossover for meme evolution can further improve the convergence speed of the algorithm, effectively avoid reducing local optimal solutions, and improve the breadth and depth of the search. The detailed steps of SFLA are as follows:

Step 1: Frog population initialization. Randomly generate a population of S with N frogs.

Step 2: Frog population sorting. Sort the frog individuals in the population using non-dominated sorting and crowding distance, and record the frog F_g with the best fitness in population S.

Step 3: Meme grouping. The grouping of individual frogs is equivalent to a non-dominated ranking hierarchy, and the optimal solutions from each hierarchy are selected to form a meme group.

Step 4: Meme evolution. Independently perform meme evolution on the meme group and update the worst frog in the group.

Step 5: Meme group reconstruction. Mixed cultural genomes: after completing a local search in each meme group, reassemble population S to complete communication and communication between frogs, reorder according to step 2, update the optimal frog in each meme group, and record the frog F_g with the best fitness in population S.

Step 6: If the maximum evolution number (Gen) or convergence condition of the entire population has not been reached, return to step 3 to continue. Otherwise, stop the algorithm iteration and output the global optimal value.

The basic process of the improved shuffled frog leaping algorithm in this article is shown in Figure 5.

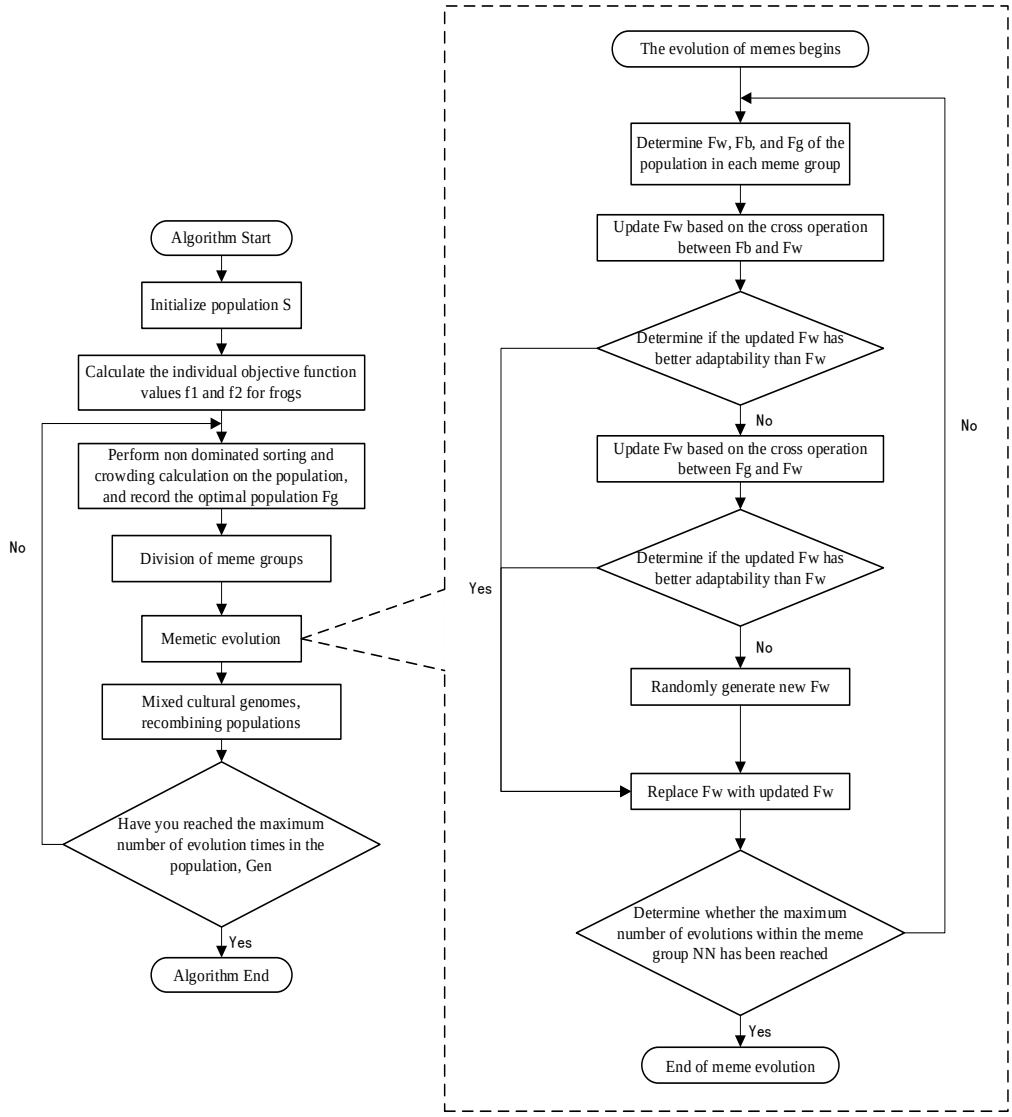

Figure 5. Flow diagram of improved SFLA.

5. Numerical Tests

To verify the rationality and applicability of the improved hybrid leapfrog algorithm designed in this article in solving the order-scheduling problem considering worker cooperation and learning effects in the seru production environment, MATLAB R2022a software was used to conduct experimental simulation verification on large-scale and small-scale examples.

The parameters set in Table 1 are for using the improved hybrid leapfrog algorithm to solve specific examples in this article. The experimental parameter settings of the size calculation examples in this article refer to the experimental settings of Lian et al.'s [1] calculation examples, and adjustments have been made based on this. In the production process of seru, due to different product types, the corresponding processes and standard processing times of the products may vary. The specific processes and standard processing times of the products are shown in Table 2. The specific processing skills of workers are shown in Table 3.

Small scale examples: Assuming there are 4 seru units and 12 workers in the production workshop, and 10 orders have arrived, order scheduling and allocation are required. The basic information on the orders is shown in Table 4. This article allocates orders based on the size of their delivery dates. The order of order allocation is Order 6, Order 1, Order 5, Order 7, Order 9, Order 4, Order 10, Order 2, Order 8, and Order 3.

Table 1. Example parameters.

Systemic Factors	Parameter Value
pop_size	100
m_frog	20
n_frog	5
Gen	50
a_{wu}	N(0, 0.01)
M	U(0, 1)
b	U(−1, 0)
worker_lowlim	2
worker_uplim	5
T_seru	2400

Table 2. Standard processing time for the corresponding process of the product.

	S1	S2	S3	S4	S5	S6	S7	S8
P1	2.92	2.79	3.36	-	3.72	2.94	3.12	3.08
P2	-	2.79	3.36	3.40	3.72	-	3.12	3.08
P3	2.92	2.79	3.36	3.40	3.72	2.94	3.12	3.08

Note: "-" indicates that the product does not require this processing step.

Table 3. The corresponding process for workers' processing skills.

	S1	S2	S3	S4	S5	S6	S7	S8
W1	√	-	√	-	√	√	√	√
W2	-	√	√	√	√	√	-	√
W3	√	√	-	√	√	-	√	√
W4	-	√	√	√	√	-	√	√
W5	√	√	√	√	√	-	√	-
W6	-	√	√	-	√	√	√	√
W7	√	√	√	-	-	√	√	√
W8	-	√	√	√	√	-	√	√
W9	√	√	√	√	√	√	√	√
W10	-	√	√	√	√	-	√	√
W11	√	√	√	-	√	√	√	√
W12	-	√	√	√	√	-	√	√

Note: "√" indicates that the worker has mastered the skill, "-" Indicates that the worker does not master the skill.

Table 4. Example of order basic information.

Order Number	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10
Product type	1	3	2	2	3	2	1	1	2	2
Order	32	16	17	28	17	35	35	33	19	31
Delivery date	750	1200	1800	900	780	550	850	1750	870	960
Setup time	3.4	2.7	3.1	3.1	2.7	3.1	3.4	3.4	3.1	3.1

To verify the necessity of considering the effects of worker cooperation and learning behavior when solving order-scheduling problems in this article, a comparative experiment was conducted.

Example experiment 1: Research the necessity of considering the effects of learning behavior. The indicators for considering learning behavior are selected with an incompressible factor of M = 0.79 and a learning index of b = −0.7. Other variables and parameters are controlled to be consistent with the control group. In the control group, worker cooperation and learning behavior were not considered. The seru order-scheduling problem with and without considering worker learning effects is solved separately, and the results are shown in Figure 6. From the results, it can be seen that under the premise of this article, the order-scheduling scheme considering worker-learning effects obtains better solutions for

the maximum completion time and total processing time of worker orders in the seru production system; that is, considering worker-learning effects in the seru scheduling scheme can obtain better allocation results.

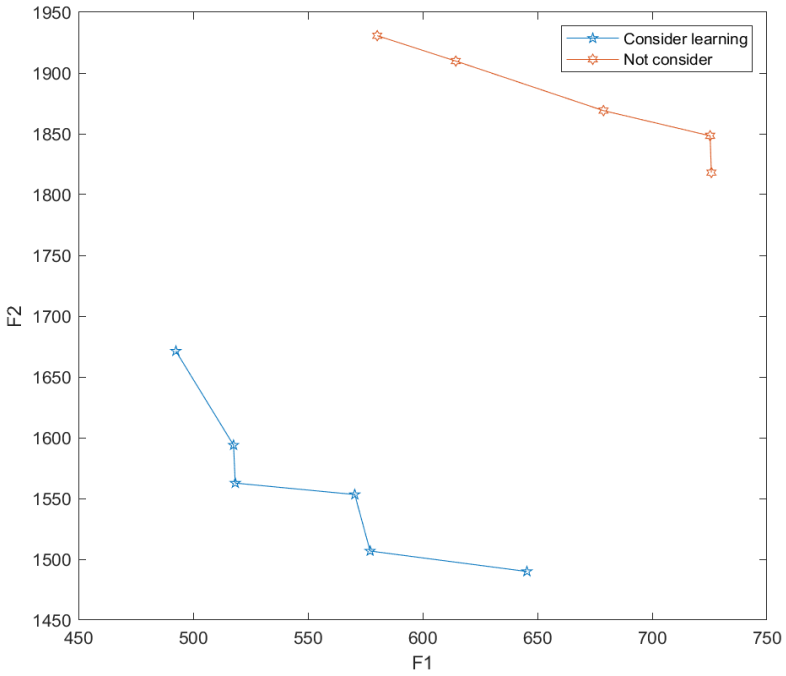

Figure 6. Comparison of results in example experiment 1.

When considering the learning effect of workers, this article adopts the DeJong learning effect model. In the DeJong learning effect model, changes in the incompressible factor M and learning index b will both affect the learning effect. To further explore the impact of worker learning effects on order scheduling, this article changes the incompressible factor and learning index that affect learning effects.

Example experiment 2: Changing the incompressibility factor and learning index. Selecting incompressible factors M with values of 0.79, 0.89, and 0.95, and controlling for the same other variables, the results obtained are shown in Figure 7. From the results of Figure 7, it can be seen that when the learning index remains unchanged and the incompressibility factor increases, the solutions obtained for both objective functions increase. The incompressible factor represents the impact of the automation level of the production line on the learning effect. A larger index indicates a greater degree of automation in the assembly line and a diminished learning ability of workers. This indicates that in the seru production system, an appropriate level of automation can improve the completion efficiency of orders. However, since the seru production system mainly relies on worker production, a higher level of automation is not conducive to the improvement in worker learning ability and affects the production time of orders. Selecting learning indices of -0.7, -0.8, and -0.9, controlling for the same other variables, the results are shown in Figure 8. From the results of Figure 8, it can be seen that when the control incompressibility factor remains constant and the learning index $|\ b\ |$ is increased, the overall learning ability continues to increase, and the solutions obtained from the two objective functions tend toward the origin; that is, as the learning index $|\ b\ |$ increases, the results become better. This further demonstrates the necessity of considering the worker-learning effect in order scheduling for a reasonable arrangement of order scheduling.

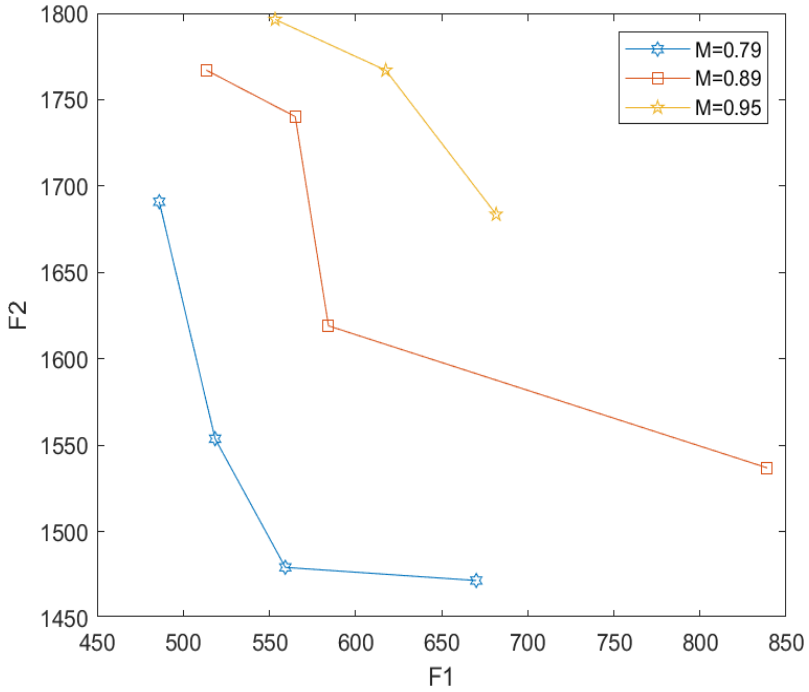

Figure 7. Comparison of results with different incompressibility factors in example experiment 2.

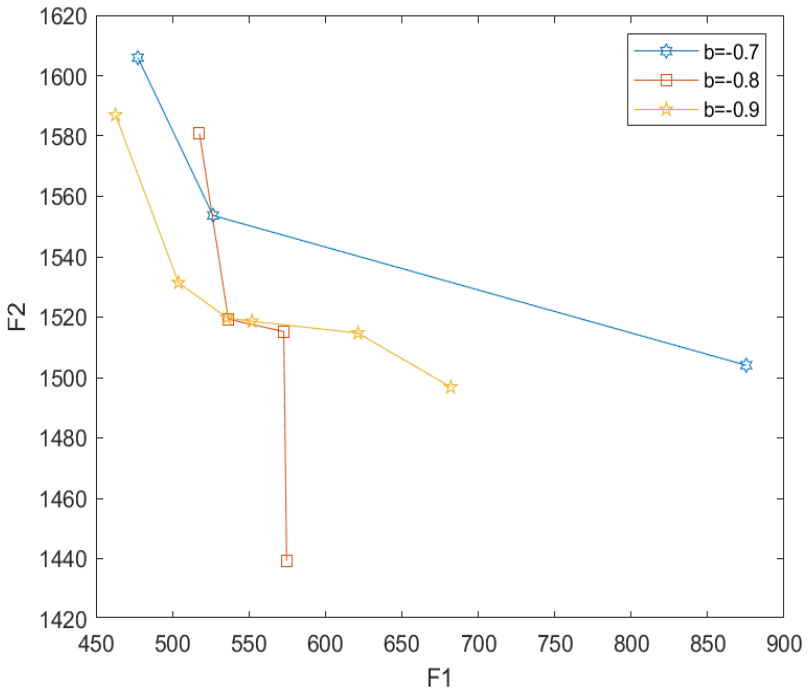

Figure 8. Comparison of results of different learning indices in example experiment 2.

Example experiment 3: Research on the necessity of considering the effects of worker cooperative behavior. The cooperation coefficient of randomly selected workers remains unchanged in the comparative experiment, and other variables and parameters are controlled to be consistent with the control group. In the control group, worker cooperation and learning behavior were not considered. The results are shown in Figure 9. In Figure 9, it can be seen that the order-scheduling scheme obtained by considering the cooperative behavior of workers has a better solution for the objective function of maximum completion time and total order processing time. Therefore, considering the cooperative behavior of workers is beneficial for the allocation of seru order scheduling.

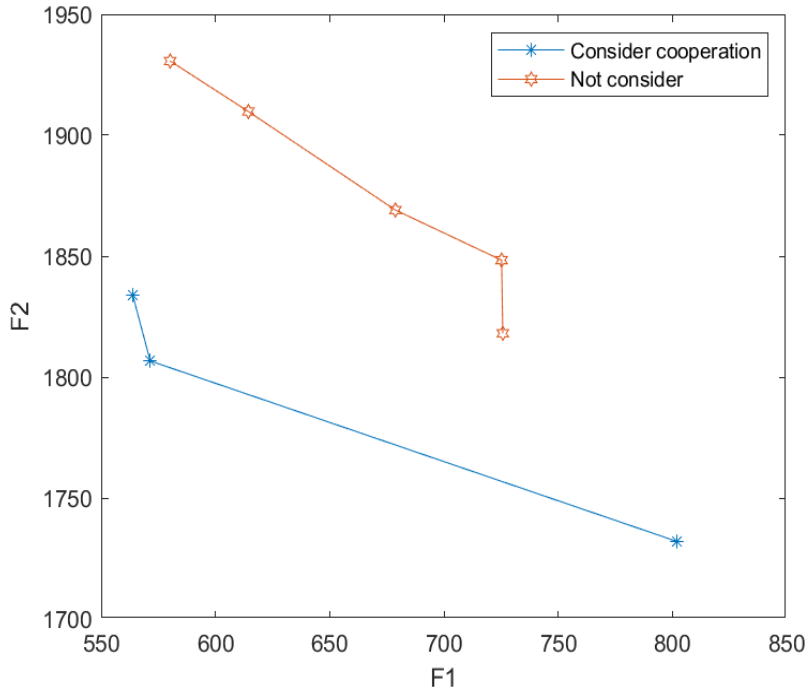

Figure 9. Comparison of results in example experiment 3.

To further explore the impact of worker behavior effects on seru scheduling and to consider the dual factors of worker cooperation and learning behavior effects on seru order scheduling, case experiment 4 was conducted.

Example experiment 4: The necessity of considering the dual behavioral effects of workers was studied, and the experimental results are shown in Figure 10. In Figure 10, it can be seen that the optimal solutions obtained by considering worker behavior effects are better than those obtained without considering worker behavior effects, indicating that considering worker behavior effects can improve the efficiency of the seru production system. In Figure 10, it can also be observed that the optimal solution obtained by considering the dual behavioral effects of workers is superior to the optimal solution obtained by solely considering the cooperative or learning behavioral effects of workers. This indicates that in actual production scheduling, considering both cooperative and learning behaviors of workers to develop order-scheduling schemes can better leverage the effects of worker behavior, maximize human production efficiency, and optimize the performance of the seru system.

To verify the effectiveness of the algorithm, this paper applies the GA algorithm and the improved shuffled frog leaping algorithm to solve the problem through small-scale examples. To provide a more intuitive reflection of the results, this article has expanded the production volume of orders. The same applies to other items, and Table 5 shows the order demand quantity table. Figure 11 shows the results obtained from running three algorithms five times. The GSFLA algorithm applies a crossover operator based on the traditional shuffled frog leaping algorithm. The crossover operator is the same as in this article, but the meme group evolution is different. The GSFLA algorithm applies the meme group evolution idea of the traditional shuffled frog leaping algorithm. In the graph, it can be seen that the algorithm in this study is superior to the GSFLA algorithm but has no significant advantage compared to the GA algorithm. However, the average single operation time of the algorithm in this paper is 0.95 s, and the average single operation time of the GA algorithm is 16.2 s, reducing the running time by 94%. Through the above comparison, it can be concluded that the algorithm in this study is effective, but there are still shortcomings in improvement. In the future, the idea of the genetic algorithm can be further referenced to improve the optimization ability.

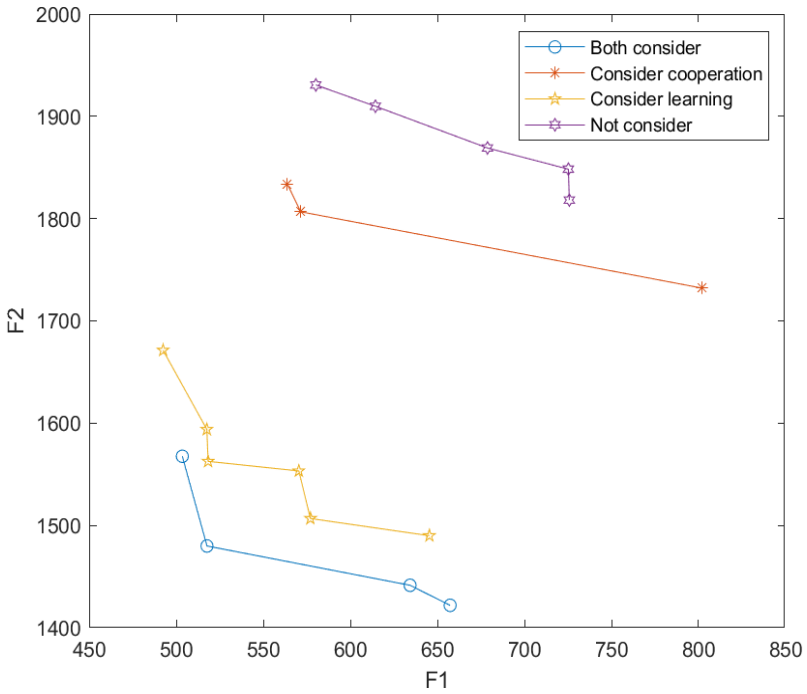

Figure 10. Comparison of results in example experiment 4.

Table 5. Order quantity information.

Order Number	J1	J2	J3	J4	J5	J6	J7	J8	J9	J10
Order	164	115	117	128	127	135	185	133	119	131

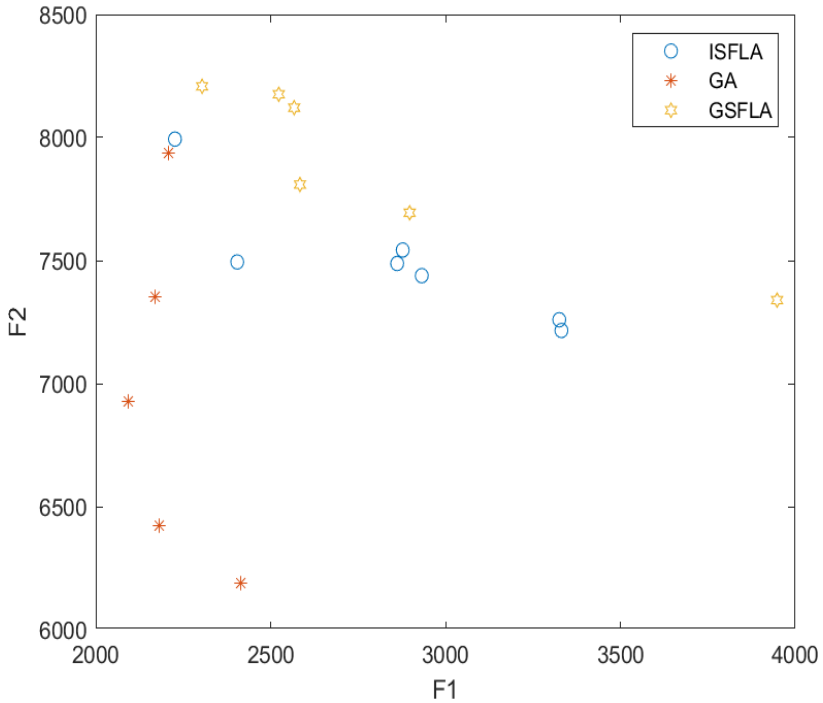

Figure 11. Non-dominated solution sets of three algorithms.

In the actual seru production scenario, the batch size of orders received by the workshop will be greater than that in small-scale examples, and the relationship between seru units and orders will also be more complex.

To further validate the effectiveness of the algorithm, this article intends to conduct large-scale numerical experiments, with specific experimental parameters shown in Table 6.

Table 6. Parameter information of large-calculation examples.

Parameter	Value Range
Seru	6
Worker	18
Order	25
Order quantity	[20, 40]
Workers' master skills	[6, 8]
Delivery lead time	[800, 2000]
Incompressible factor M	0.79
Learning Index b	−0.7

Table 7 shows the non-dominated solution set for the large-scale experiment. Table 8 and Figure 12 take the large example of non-dominant solution 1 as an example and list the Gantt charts of the optimal solution for the optimal solution and the optimal order-scheduling scheme. In summary, in the design process of production scheduling, enterprises can improve the efficiency of the seru production system by leveraging worker cooperation and learning behavior.

Table 7. Large example experiment's non-dominant solution set.

F1	F2
1182.22	5567.60
1203.59	4722.48
1483.43	4496.41
1524.19	4365.36
2084.35	4295.84

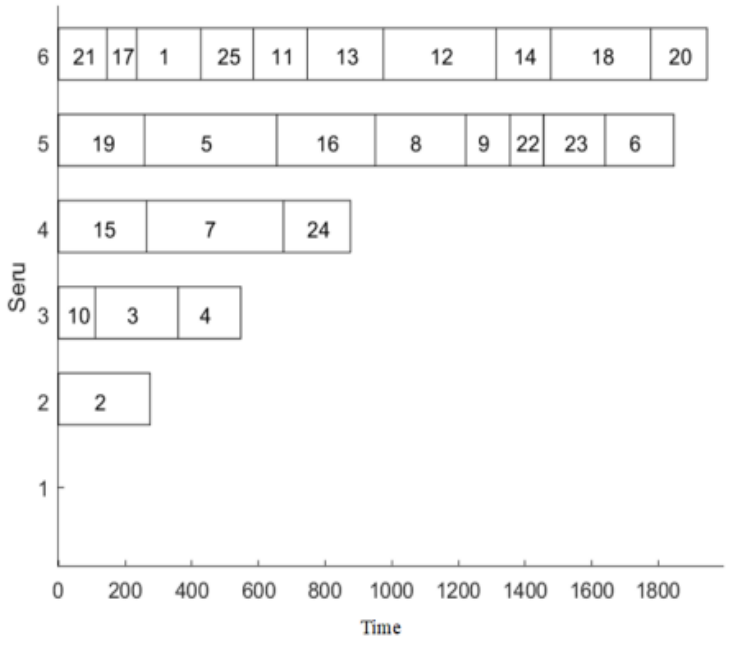

Figure 12. Gantt chart of order scheduling results in large-example experiments. Note: the numbers on the graph represent the order number.

Table 8. Large-example worker-allocation plan.

Worker	Seru	Worker	Seru
1	3	10	3
2	1	11	3
3	4	12	6
4	1	13	2
5	6	14	6
6	5	15	4
7	5	16	6
8	5	17	2
9	4	18	3

6. Conclusions

This article studies the order-scheduling problem while considering the effects of worker cooperation and learning behavior. In constructing the problem model, factors such as worker cooperation behavior, learning behavior, and preparation time for order processing are taken into account. A nonlinear programming model is established to minimize the maximum completion time and order processing time. To better solve the seru production scheduling problem, this paper introduces a hybrid frog jump algorithm with a strong global optimization ability to solve it. Based on the order-scheduling problem solved in this article and the characteristics of the model, the hybrid leapfrog algorithm is improved. The effectiveness of the model and algorithm has been demonstrated through small-scale and large-scale numerical experiments. The following conclusions can be drawn from the example experiment: (1) Both cooperative and learning behaviors of workers will have an impact on seru production, and learning behavior has a greater impact on seru production efficiency than cooperative behavior. (2) Considering the dual behavioral effects of workers can optimize the seru's production efficiency. (3) Positive cooperative behavior can promote seru production, but negative cooperative behavior can hinder seru production. Based on the aforementioned findings, we can offer several recommendations for business managers. Firstly, enterprises should focus on creating a conducive work environment, enhancing employee cooperation awareness, and facilitating communication and collaboration to improve workshop production efficiency. Secondly, when developing an order-scheduling plan, managers should take into account the learning tendencies of workers during the order-production process. They should also fully utilize the learning behavior effects of workers, gathering similar orders to improve worker skill proficiency and further improving the efficiency of seru production systems.

The order-scheduling problem addressed in this article, with known order requirements, does not consider the impact of dynamic factors such as random order arrival and new order arrival. It is limited to static order scheduling. Future research can focus more on dynamic factors and consider random order demands to better fit the actual seru production environment.

Author Contributions: Conceptualization, W.L. and W.D.; methodology, W.L. and W.D.; validation, W.D. and X.W.; formal analysis, W.D. and X.W.; investigation, W.D. and X.W.; data curation, W.D.; writing—original draft preparation, W.D.; writing—review and editing, W.L. and W.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Natural Science Regional Foundation of China (72061029 and 71861031).

Data Availability Statement: Data are contained within the article.

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

Notation

i	Index of seru, $i = 1, 2, \ldots, I$.
j	Index of order, $j = 1, 2, \ldots, J$.
w	Index of worker, $w = 1, 2, \ldots, W$.
p	Product types, $p = 1, 2, \ldots, P$.
k	Processing sequence of orders in seru, $k = 1, 2, \ldots, K$.
T_p^s	Standard processing time for process s of product p.
M	Incompressible factor ($0 \leq M \leq 1$).
b	Learning index ($-1 \leq b \leq 0$).
α_{wu}	Cooperation coefficient between worker w and worker n.
α_w	Cooperative influence factors of workers w.
S_j	Quantity of order j.
W_{min}	The lower limit for worker allocation within seru.
W_{max}	The upper limit for workers allocated within seru.
FCB_j	Start time of order j in seru i.
ST_j	Preparation time of order j in seru i.
FC_j	Processing time of order j in seru i.
Q_{jw}	Number of orders completed by worker w.
Q_{ijw}	The number of orders completed by worker w in seru i.
T_{jw}^s	The total processing time for worker w to complete process s in order j.
T_i	Available production time for seru i.
Z_p	Replacement time for product p.
D_j	Delivery date of order j.

$$Y_{wi} = \begin{cases} 1, \ if \ worker \ w \ is \ assigned \ to \ seru \ i \\ 0, \ otherwise \end{cases}$$

$$R_{ij} = \begin{cases} 1, \ if \ order \ j \ is \ processed \ in \ seru \ i \\ 0, \ otherwise \end{cases}$$

$$C_{jp} = \begin{cases} 1, \ if \ the \ product \ type \ of \ order \ j \ is \ p \\ 0, \ otherwise \end{cases}$$

$$X_{ijk} = \begin{cases} 1, \ if \ the \ processing \ position \ of \ order \ j \ in \ seru \ i \ is \ k \\ 0, \ otherwise \end{cases}$$

$$U_{wp} = \begin{cases} 1, \ If \ the \ type \ of \ product \ processed \ by \ worker \ w \ is \ p \\ 0, \ otherwise \end{cases}$$

References

1. Lian, J.; Liu, C.; Yin, Y. A multifunctional worker allocation model and algorithm considering worker heterogeneity under seru production mode. *Oper. Res. Manag.* **2019**, *28*, 81–89.
2. Zeng, S.; Wu, Y.; Yu, Y. Research on fairness oriented joint decision making of multifunctional work batch job allocation in seru production system. *Ind. Eng. Manag.* **2023**, *28*, 58–65.
3. Zhang, X.; Liu, C.; Li, W.; Evans, S.; Yin, Y. Effects of key enabling technologies for seru production on sustainable performance. *Omega* **2017**, *66*, 290–307. [CrossRef]
4. Liu, C.; Stecke, K.E.; Lian, J.; Yin, Y. An implementation framework for seru production. *Int. Trans. Oper. Res.* **2014**, *21*, 1–19. [CrossRef]
5. Treville, S.; Ketokivi, M.; Singhal, V.R. Competitive manufacturing in a high-cost environment: Introduction to the special issue. *J. Oper. Manag.* **2017**, *49–51*, 1–5. [CrossRef]
6. Yamada, H. *Waste Reduction*; Gentosha: Tokyo, Japan, 2009. (In Japanese)
7. Yu, Y.; Tang, J. Review of seru production. *Front. Eng. Manag.* **2019**, *6*, 183–192. [CrossRef]
8. Yin, Y.; Stecke, K.E.; Swink, M.; Kaku, I. Lessons from seru production on manufacturing competitively in a high cost environment. *J. Oper. Manag.* **2017**, *49*, 67–76. [CrossRef]
9. Zhan, R.; Zhang, J.; Cui, Z.; Peng, J.; Li, D. An automatic heuristic design approach for seru scheduling problem with resource conflicts. *Discret. Dyn. Nat. Soc.* **2021**, *2021*, 8166343. [CrossRef]
10. Jiang, Y.; Zhang, Z.; Gong, X.; Yin, Y. An exact solution method for solving seru scheduling problems with past-sequence-dependent setup time and learning effect. *Comput. Ind. Eng.* **2021**, *158*, 107354. [CrossRef]
11. Li, D.; Jiang, Y.; Zhang, J.; Cui, Z.; Yin, Y. An on-line seru scheduling algorithm with proactive waiting considering resource conflicts. *Eur. J. Oper. Res.* **2023**, *309*, 506–515. [CrossRef]
12. Wu, Y.; Wang, L.; Chen, J.F.; Zheng, J.; Pan, Z. A reinforcement learning driven two-stage evolutionary optimisation for hybrid seru system scheduling with worker transfer. *Int. J. Prod. Res.* **2023**, 1–20. [CrossRef]

13. Lian, J.; Li, W.; Pu, G.; Zhang, P. Bi-objective energy-efficient scheduling in a seru production system considering reconfiguration of serus. *Sustain. Comput. Inform. Syst.* **2023**, *39*, 100900. [CrossRef]
14. Zhang, X.; Zhang, Z.; Gong, X.; Yin, Y. An exact branch-and-bound algorithm for seru scheduling problems with sequence-dependent setup time. *Soft Comput.* **2023**, *27*, 6415–6436. [CrossRef]
15. Shen, L.; Zhang, Z.; Song, X.; Yin, Y. A hybrid GA-PSO algorithm for seru scheduling problem with dynamic resource allocation. *Int. J. Manuf. Res.* **2023**, *18*, 100–124. [CrossRef]
16. Zhang, Z.; Song, X.; Huang, H.; Yin, Y.; Lev, B. Scheduling problem in seru production system considering DeJong's learning effect and job splitting. *Ann. Oper. Res.* **2022**, *312*, 1–23. [CrossRef]
17. Zhang, Z.; Shen, L.; Gong, X.; Zhong, X.; Yin, Y. A genetic-simulated annealing algorithm for stochastic seru scheduling problem with deterioration and learning effect. *J. Ind. Prod. Eng.* **2023**, *40*, 205–222. [CrossRef]
18. Zhang, Z.; Song, X.; Gong, X.; Yin, Y.; Lev, B.; Zhou, X. An effective heuristic based on 3-opt strategy for seru scheduling problems with learning effect. *Int. J. Prod. Res.* **2023**, *61*, 1938–1954. [CrossRef]
19. Jiang, Y.; Zhang, Z.; Song, X.; Yin, Y. seru scheduling problems with multiple due-windows assignment and learning effect. *J. Syst. Sci. Syst. Eng.* **2022**, *31*, 480–511. [CrossRef]
20. Dong, P.; Yu, J. A Production Scheduling Model for Clothing Sewing Workshop Considering Learning Forgetting Effect. *Mod. Text. Technol.* **2023**, *31*, 81–91.
21. Janiak, A.; Kovalyov, M.Y.; Lichtenstein, M. Strong NP-hardness of scheduling problems with learning or aging effect. *Ann. Oper. Res.* **2013**, *206*, 577–583. [CrossRef]
22. Sun, L.H.; Cui, K.; Chen, J.H.; Wang, J.; He, X.C. Some results of the worst-case analysis for flow shop scheduling with a learning effect. *Ann. Oper. Res.* **2013**, *211*, 481–490. [CrossRef]
23. Biskup, D. A state-of-the-art review on scheduling with learning effects. *EuropeanJournal Oper. Res.* **2008**, *188*, 315–329. [CrossRef]
24. Wright, T.P. Factors affecting the cost of airplanes. *J. Aeronaut. Sci.* **1936**, *3*, 122–128. [CrossRef]
25. Carr, G.W. Peacetime cost estimating requires new learning curves. *Aviation* **1946**, *45*, 220–228.
26. Asher, H. Cost-Quantity Relationships in the Airframe Industry. Ph.D. Thesis, The Ohio State University, Columbus, OH, USA, 1956.
27. Dejong, J.R. The effects of increasing skill on cycle time and its consequences for time standards. *Ergonomics* **1957**, *1*, 51–60. [CrossRef]
28. Baloff, N. Extension of the learning curve—Some empirical results. *J. Oper. Res. Soc.* **1971**, *22*, 329–340. [CrossRef]
29. Zhang, H.; Xu, G.; Zhang, J. Research on multi-stage hybrid flow shop scheduling problem with learning effects. *J. Chongqing Norm. Univ. (Nat. Sci. Ed.)* **2021**, *38*, 87–95.
30. Hu, J.; Wu, Y.; Wang, Y.; Wu, Y. A Single Person Job Shop Scheduling Algorithm Considering Learning Effects. *Control Decis. Mak.* **2022**, *37*, 37–46.
31. Wang, J.; Liu, C.; Zhou, M. Improved bacterial foraging algorithm for cell formation and product scheduling considering learning and forgetting factors in cellular manufacturing systems. *IEEE Syst. J.* **2020**, *14*, 3047–3056. [CrossRef]
32. Sakamaki, H. *Canon Cellular Production Method*; Oriental Press: Beijing, China, 2006.
33. Wang, Y.; Tang, J. A Method for Constructing Unit Assembly Systems Considering Employee Collaboration. *Control Decis. Mak.* **2020**, *35*, 453–460.
34. Cao, H.; Kong, F. Collaborative simulation of U-shaped assembly line workers based on emotional model. *Comput. Integr. Manuf. Syst.* **2015**, *21*, 3209–3221.
35. Liu, F.; Fang, K.; Tang, J.; Yin, Y. Solving the rotating seru production problem with dynamic multi-objective evolutionary algorithms. *J. Manag. Sci. Eng.* **2022**, *7*, 48–66. [CrossRef]
36. Li, X.; Zhang, Z.; Sun, W.; Liu, Y.; Tang, J. Parallel dynamic NSGA-II with multi-population search for rescheduling of seru production considering schedule changes under different dynamic events. *Expert Syst. Appl.* **2023**, *238*, 121993. [CrossRef]
37. Miao, Q.; Bai, Z.; Liu, X.; Awais, M. Modelling and numerical analysis for seru system balancing with lot splitting. *Int. J. Prod. Res.* **2023**, *61*, 7410–7433. [CrossRef]
38. Pan, C.; Yan, H.; Zhan, Y.; Zhang, W. seru production scheduling based on multi-population multi-objective genetic algorithm. *Control Eng.* **2023**, *30*, 1567–1574.
39. Yilmaz, B.G.; Yilmaz, Ö.F.; Çevikcan, E. Lot streaming in workforce scheduling problem for seru production system under Shojinka philosophy. *Comput. Ind. Eng.* **2023**, *185*, 109680. [CrossRef]
40. Eusuff, M.M.; Lansey, K.E. Optimization of water distribution network design using the shuffled frog leaping algorithm. *J. Water Resour. Plan. Manag.* **2003**, *129*, 210–225. [CrossRef]
41. Cao, J.; Wang, L.; Lei, D. Distributed assembly hybrid flow shop scheduling based on frog jump algorithm. *J. Huazhong Univ. Sci. Technol. (Nat. Sci. Ed.)* **2023**. [CrossRef]
42. Meng, L.; Zhang, B.; Ren, Y.; Zhang, C. A Hybrid Frog Jump Algorithm for Solving Distributed Flexible Job Shop Scheduling. *J. Mech. Eng.* **2021**, *57*, 263–272.