Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 6357394, 12 pages
https://doi.org/10.1155/2020/6357394

Research Article

Research on Construction and Application for the Model of Multistage Job Shop Scheduling Problem

Jianzhong Xu, Song Zhang, and Yuzhen Hu

School of Economics and Management, Harbin Engineering University, Harbin 150001, Heilongjiang, China

Correspondence should be addressed to Song Zhang; zhangsong@hrbeu.edu.cn

Received 1 April 2020; Revised 27 May 2020; Accepted 1 June 2020; Published 6 July 2020

Academic Editor: Elena Zaitseva

Based on the practical application of an enterprise, we address the multistage job shop scheduling problem with several parallel machines in the first stage (production), a few parallel machines in the second stage (processing and assembly), and one machine in the following stages (including joint debugging, testing, inspection, and packaging). First, we establish the optimization objective model for the first two stages. Then, based on the design of the sequencing algorithm in the first two stages, a correction algorithm is designed between the first stage and the second stage to solve this problem systematically. Finally, we propose two benchmark approaches to verify the performance of our proposed algorithm. Verification of numerical experiments shows that the model and algorithm constructed in this paper effectively improve the production efficiency of the enterprise.

1. Introduction

Multistage scheduling problems are common in the actual production scheduling of enterprises. Since the beginning of the 21st century, with the rapid development of the social economy, the diversified and personalized needs of customers have been increasing daily. The relationship between market supply and demand is gradually changing from a seller's market to a buyer's market, and competition among enterprises is intensifying, forcing enterprises to change their production mode towards diversification, multiple varieties, and small batches. Therefore, how to respond quickly, optimize production, and meet customers' needs efficiently and punctually has become the core factor of competition among enterprises. In this context, most enterprises modularize their production according to process characteristics to form a number of production and assembly workshops with specific functions. At this time, the multistage scheduling problem arises in production scheduling. However, most existing research remains theoretical, with few studies on practical application, so there is an urgent need to establish a set of system methods that can be applied to the multistage production enterprise production scheduling mode.

In this study, we focus on the multistage job shop scheduling problem. The constraints (such as processing time and processing sequence) in the process of multistage job shop scheduling are considered to construct a mathematical model for different stages of production scheduling, and both a multistage separated production scheduling algorithm and an improved multistage relevance production scheduling algorithm are designed to solve the proposed problem. Finally, the improved multistage relevance production scheduling algorithm is verified through a computational experiment based on real-world data. Two contributions are made in this paper: (1) in the process of calculating the solution, the theory of scheduling and a heuristic algorithm are combined to obtain a satisfactory multistage production scheduling solution; (2) compared with previous articles, this paper pays more attention to the practical application of mathematical problems in enterprises. It not only improves the production efficiency of multistage manufacturing enterprises but also provides a scientific reference for the production decision-making of enterprises. Furthermore, the approach can create more profits for enterprises.

The remainder of this paper is organized as follows. Section 2 reviews the relevant literature on production scheduling. Section 3 builds a model for the multistage job shop scheduling problem. Section 4 designs the corresponding algorithm for the multistage job shop scheduling mode. Section 5 reports the numerical experiments of the

model of multistage job shop scheduling, and conclusions are drawn in Section 6. The findings can be used as a reference for production enterprises' decision-making and have good application value.

2. Literature Review

Production scheduling, which can be defined as to "allocate limited resources to several tasks in time to meet or optimize one or more objectives" [1], is a key task for manufacturing enterprises in implementing production plans. Moreover, production scheduling is an important field in operational research. Since Johnson [2] proposed an effective optimization algorithm for solving N/2/F/Cmax and some special n/3/F/Cmax problems, many scholars have engaged in production scheduling research. Graves [3] systematically summarized the types of production scheduling based on previous research. According to the processing complexity, he divided production scheduling into four types: one processor, parallel processors, flow shop, and job shop. This classification method has been adopted and quoted by later scholars.

Many scholars engaged in one-processor scheduling and parallel-processor scheduling research in the early stage [4–8], and many scholars are continuing research in this field [9–15]. Luo et al. [16] studied the single-machine scheduling problem with job-dependent machine deterioration. Chen and Yuan [17] studied the classic single-machine scheduling problem with deadlines based on minimizing the total tardiness and proved that the problem is NP-hard. Luo and Liu [18] focused on the single-machine scheduling problem with workload dependent on maintenance time and extended the previous theoretical results. Atakan et al. [19] studied the single-machine scheduling problem with uncertain parameters. Zhang et al. [20] proposed an unrelated parallel machine scheduling problem based on the consideration of processing speed and processing time, and they verified the effectiveness of their proposed heuristic evolutionary algorithm. Vallada et al. [21] studied the problem of unrelated parallel machines with one additional resource using a scatter search algorithm. Alidaee et al. [22] used dynamic programming to study a parallel machine scheduling model in which the job processing rates are interdependent and jobs have no priority. Vincent et al. [23] studied the capacity-constrained batch problem with multiple projects, set time and unrelated parallel machines, and compared the results with those of Toledo and Armentano [24] to demonstrate the competitiveness of the existing research results.

Meanwhile, scholars represented by Johnson [2] are also studying the multistage production scheduling problem. The development of production modularization and integration in manufacturing enterprises has led to complex relations among production processes, which often affect the progress of the whole production system. Indeed, the multistage flow shop scheduling and multistage job shop scheduling are more in line with the actual needs of enterprises. Multistage flow shop scheduling is applicable to steel manufacturing [25, 26], printed circuit boards [27], automobile manufacturing [28], and other enterprises whose products must be produced and processed according to a certain technological process. Multistage job shop scheduling is more common in manufacturing enterprises because of its uncertain processing process [29–32]. However, due to the lack of practical application background, most scholars are engaged in research from the perspective of improving the speed [33, 34] and solution quality [35, 36] of multistage production scheduling algorithms. Samarghandi [37] studied the problem of minimizing the maximum completion time of flow shop scheduling under the constraints of a minimum and maximum time delay. By constructing a mixed-integer linear programming model and a constraint programming model, the author proposed a tabu search algorithm and verified the effectiveness of the proposed tabu search algorithm with an example. Gong et al. [38] proposed a hybrid evolutionary algorithm based on the flexibility of workers that considered factors such as energy consumption. Experiments on benchmark examples demonstrated that the proposed algorithm was significantly better than two other well-known algorithms in terms of solution quality and computational efficiency. Caldeira and Gnanavelbabu [39] used an improved Java algorithm to solve a flexible job shop scheduling problem and compared the solution quality with other famous metaheuristic algorithms to verify the superiority of their algorithm. Dabah et al. [40] paralleled the tabu search algorithm to solve the blocking job shop scheduling problem and used data in the existing literature to verify that the designed algorithm can improve upon the best results achieved by a large number of benchmark methods in the literature. Zhang et al. [41] proposed an improved hybrid particle swarm optimization algorithm to study the multi-objective flexible job shop scheduling problem. The benchmark experiment showed that the algorithm has the advantages of high quality and fast convergence.

A literature review finds many research results related to the production scheduling problem. However, the production scheduling problem is characterized by not only many production constraints but also a large number of production sequencing combinations, so a general production scheduling theory is difficult to develop. Existing scholars focus on improving the solution quality of the algorithms and give less attention to the application of the production scheduling model to solve the actual problems of enterprises. Therefore, the application of multistage production scheduling to better serve real manufacturing enterprises must be studied.

3. Problem Description and Model Formulation

3.1. Problem Description. Quality, cost, and delivery time are the three core issues in the production and operation management of enterprises. This paper focuses on the delivery time to solve a production scheduling problem in which the products go through multiple stages of production before final delivery. In this multistage production, the first two stages are usually the most complex because the product processing techniques of these two stages, which are the key factors affecting the production efficiency, are both independent and influence each other. The subsequent stages are all single-machine operations, so we simply need to sum the standard working hours of each stage. To summarize, we take the first two stages of the production scheduling process as the key content to be studied.

3.1.1. Problem Description of the First-Stage Production Scheduling. The first stage usually involves producing intermediate products and providing raw materials for subsequent assembly. Let j workstations produce i parts in this stage. In this process, due to the limited number of processing workstations, there is no specific requirement for prioritizing which type of intermediate products to produce, which leads to many plans for production. However, the processing time for different types and models of intermediate products is different, and intermediate products can be processed at any workstation. At any time, an intermediate product can be processed at most at one workstation, and each workstation can process only one intermediate product at a time. The processing of intermediate products in any processing stage is not allowed to be interrupted. This scenario inevitably leads to different processing times for each workstation to complete the intermediate products. For a complete set of products, the production schedule of the first stage directly affects the assembly time of the second stage, so the production of the first stage must be scheduled carefully.

3.1.2. Problem Description of the Second-Stage Production Scheduling. The second stage is the process of product system integration, and the speed of its progress directly affects the delivery time of the final product. Therefore, the production of the second stage must also be scheduled. In this stage, l workstation processes workpieces (denoted as set K), and the set K is divided into two subsets (i.e., $K = K_1 \cup K_2$): set K_1, in which workpieces can be processed directly without waiting time, and set K_2, in which workpieces have a waiting time. The waiting time is the total time spent in the production and conversion section of intermediate products in the first stage, and when the intermediate products are transferred to the second stage, they should be transferred in the form of subsystems.

3.1.3. Relevance between the First Stage and the Second Stage. The relevance of the first stage, the second stage, and the third stage can be seen in Figure 1.

The first stage and second stage have obvious relevance in the process of production scheduling.

First, the intermediate product production in the first stage and the no waiting time workpiece from set K_1 in the second stage can be conducted in parallel, and their completion time directly affects the start time of the third stage.

Second, the completion time of the intermediate product processing in the first stage directly affects the start time of the workpiece assembly with waiting time in the second stage.

Third, the workpiece with waiting time in the second stage must be assembled in the form of a subsystem.

Finally, only after all the workpieces are assembled can they be transferred to the third stage.

3.2. Model Formulation

3.2.1. Objective. During the modeling process, this paper makes some reasonable simplifications, including that the availability of raw materials does not affect the processing process, workstation equipment failure has little impact on processing and can be ignored, the processing time of all intermediate products is known in standard working hours, and there is no disturbance of the processing time caused by the production personnel's emotions. For the convenience of subsequent modeling, we call the processing and assembly of the second-stage workpiece the assembly task.

According to the situation of the first two stages, the optimization objective is to minimize the latest completion time of the assembly task for all assembly stations in the second stage to ensure that all tasks are completed as evenly as possible before the third stage.

3.2.2. Models

(1) Parameter definitions are as follows:

I: intermediate product set
J: intermediate product processing workstation set
K': subsystem set
I_k: intermediate product set contained in the k-th subsystem, $k \in K'$
t_i: processing time of intermediate product i on the processing workstation, $i \in I$
L: assembly workstation set
K_1: assembly task set without waiting time
K_2: assembly task set with waiting time
K: assembly task set, $K = K_1 \cup K_2$
P_{kl}: assembly time of assembly task k on assembly workstation l, $k \in K$, $l \in L$
C_k: the end of processing time when assembly task k finishes processing in the first stage of intermediate products, $k \in K$

(2) Decision variables definition is as follows:

$$X_{ij} = \begin{cases} 1, & \text{intermediate product } i \text{ is processed on process } in\ g \text{ workstation } j, i \in I, j \in J, \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

S_{ij}: start processing time of intermediate product i on processing workstation j, $i \in I$, $j \in J$.

C_{ij}: end processing time of intermediate product i on processing workstation j, $i \in I$, $j \in J$.

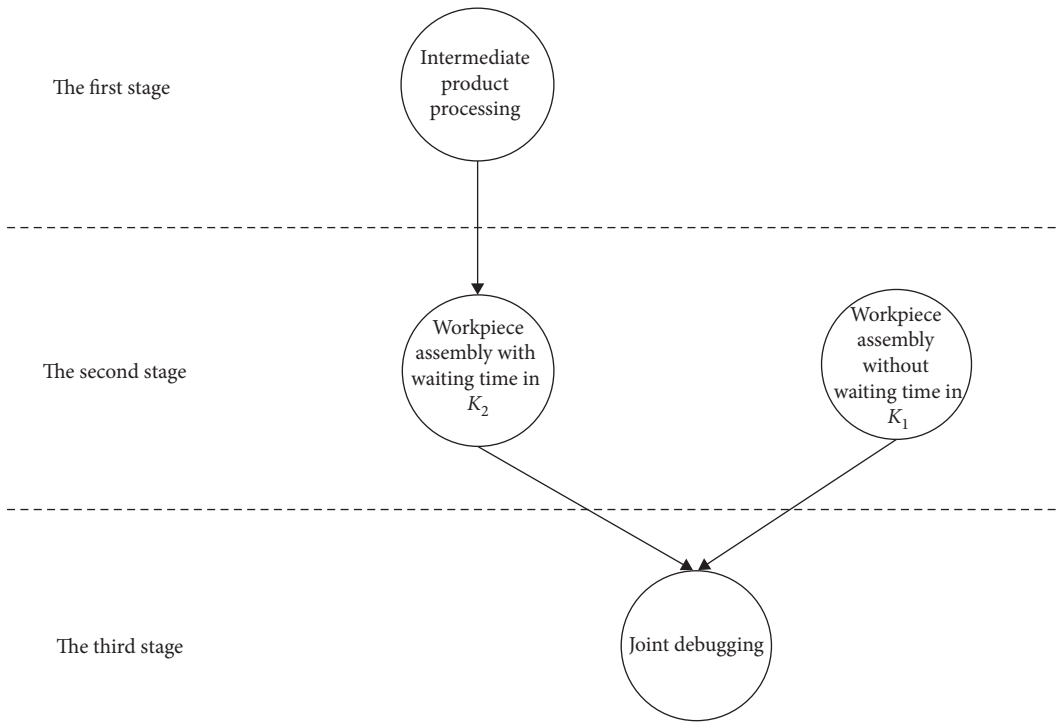

Figure 1: Relevance diagram of the first three stages.

$$X_{kl} = \begin{cases} 1, & \text{assemble task } k \text{ is assembled on assembly workstation } l, k \in K, l \in L, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

S_{kl}: start assembly time of assembly task k on assembly workstation l, $k \in K$, $l \in L$.

C_{kl}: end assembly time of assembly task k on assembly workstation l, $k \in K$, $l \in L$.

(3) Objective function is as follows:

$$\min \max_{k,l} \{S_{kl} + P_{kl}\}. \tag{3}$$

(4) Constraints are as follows:

$$\sum_{j \in J} X_{ij} = 1, \quad i \in I, \tag{4}$$

$$C_{ij} = S_{ij} + t_i, \quad i \in I,\ j \in J, \tag{5}$$

$$C_{ij}X_{ij} \le S_{i'j}X_{i'j} \text{ or } C_{i'j}X_{i'j} \le S_{ij}X_{ij}, \quad i, i' \in I,\ j \in J, \tag{6}$$

$$S_{ij} \ge 0, \quad i \in I,\ j \in J, \tag{7}$$

$$X_{ij} = 0 \text{ or } 1, \quad i \in I,\ j \in J, \tag{8}$$

$$\sum_{l \in L} S_{kl}X_{kl} > C_k, \quad k \in K, \tag{9}$$

$$\sum_{l \in L} X_{kl} = 1, \quad k \in K, \tag{10}$$

$$C_{kl}X_{kl} \le S_{k'l}X_{k'l} \text{ or } C_{k'l}X_{k'l} \le S_{kl}X_{kl}, \quad k, k' \in K, l \in L, \tag{11}$$

$$C_{kl} = S_{kl} + P_{kl}, \quad k \in K,\ l \in L, \tag{12}$$

$$S_{kl} > 0, \quad k \in K,\ l \in L, \tag{13}$$

$$X_{kl} = 0, \text{or } 1 \quad k \in K,\ l \in L. \tag{14}$$

Objective function (3) is used to minimize the latest end time of each workstation.

Constraint (4) indicates that an intermediate product can be processed by only one processing workstation. Constraint (5) represents the end processing time of intermediate product i on the processing workstation. Constraint (6) indicates that a single processing workstation cannot process two intermediate products simultaneously. Constraint (7) represents that decision variable S_{ij} is a nonnegative continuous number. Constraint (8) indicates that decision variable X_{ij} is an integer of 0 or 1. Constraint (9) indicates that the start time of the assembly task is later than the end time of the previous operation. There are two meanings: one is to take $C_k = 0$ for an assembly task that does

not need to wait; the other is to take $C_k > 0$ for an assembly task that needs to wait. Constraint (10) indicates that an assembly task can be assembled by only one assembly workstation. Constraint (11) indicates that a single assembly workstation cannot assemble two assembly tasks simultaneously. Constraint (12) shows that the end assembly time of assembly task k on assembly workstation l is the sum of its start assembly time on assembly workstation l and its assembly time on assembly workstation l. Constraint (13) represents that decision variable S_{kl} is a nonnegative continuous number. Constraint (14) indicates that decision variable X_{kl} is an integer of 0 or 1.

The above models are difficult to solve with commercial software, and enterprises prefer to obtain a satisfactory solution instead of deliberately pursuing the optimal solution when addressing actual production scheduling problems. Thus, a heuristic algorithm is good for multistage production scheduling problems. Multistage production scheduling is usually solved stage by stage, in line with the actual production situation of enterprises. Therefore, a multistage heuristic algorithm is designed in this paper.

4. Scheduling Procedure (Algorithm Design)

According to the actual situation of production, this paper focuses on the first two stages of production scheduling and designs a corresponding scheduling algorithm. Generally, the scheduling process will not start the next stage until the previous stage is completed. Based on this principle, we design two-stage algorithms in Sections 4.1 and 4.2, denoted as S1 and S2. Moreover, according to the relevance description in Section 3.1.3, the algorithm can be further optimized, so we propose an improved algorithm (denoted as S1′) for the first stage; see Section 4.3 for details.

4.1. Design of the First-Stage Production Scheduling Algorithm

4.1.1. General Idea of the Production Scheduling Algorithm for the First Stage. In the first stage, the processing sequence of intermediate products and the processing sequence of each subsystem should be determined. The general ideas for the first-stage scheduling algorithm (S1) are as follows.

First, we need to identify and summarize the intermediate products to which each subsystem belongs.

Second, because the processed intermediate products need to be transferred to the second stage in the form of a subsystem, one subsystem must be completed before processing another subsystem; that is, if the intermediate products of one subsystem are being processed, the intermediate products of another subsystem cannot be processed.

Finally, during intermediate product processing, the complexity of each subsystem is directly proportional to its processing time, so we prioritize complex subsystems, which should be transferred to the next stage as soon as possible. We give priority to complex systems, that is, tasks with long processing times, because Tang and Zhao [42] proved that the optimal solution of the maximum processing time can be obtained by performing tasks in descending order of processing time in the same speed machine scheduling problem, which will not be demonstrated here.

4.1.2. Steps of S1

Step 1: all intermediate products are classified according to their subsystems

Step 2: the processing times of all intermediate products in set I_k are added to obtain M_k, namely, $M_k = \sum_{i \in I_k} t_i$

Step 3: all subsystems are sorted in descending order according to M_k, namely, $M_k \geq M_{k+1}$, $\forall k \in \{1, \ldots, |K| - 1\}$

Step 4: all intermediate products in set I_1 are sorted in descending order of t_i, namely, $t_i \geq t_{i+1}$, $\forall i \in \{1, \ldots, |I_1| - 1\}$

Step 5: let $k = 1$. The first $|J|$ parts are taken and arranged on workstations in sequence; now, let $i = |J| + 1$

Step 6: if any workstation is processing, the i-th intermediate product in set I_k set is assigned to this workstation

Step 7: if $i > |I_k|$, then let $k = k+1$, and go to Step 8; otherwise, go to Step 6, and let $i = i+1$

Step 8: if $k < |K|$, then all intermediate products in set I_k are sorted in descending order of t_i, namely, $t_i \geq t_{i+1}$; let $i = 1$, and go to Step 6; otherwise, go to Step 9

Step 9: end

The time complexity of algorithm S1 is $O(mn^3)$, where m represents the number of processing subsystems, that is, $m = |K'|$, and n represents the maximum number of intermediate products in all subsystems, that is, $n = \max_{k \in K} |I_k|$. In Step 2, the summation of the processing time for each subsystem occurs no more than n times, so it takes at most mn iterations to obtain the total processing time for m subsystems. In Step 6, Step 7, and Step 8, intermediate product assignment of each subsystem to a workstation occurs no more than mn times. In Step 8, the complexity of the bubble sorting algorithm for each subsystem is $O(n^2)$. Therefore, the total complexity of algorithm S1 is $O(mn) + O(mn^3) = O(mn^3)$.

4.2. Design of the Second-Stage Production Scheduling Algorithm

4.2.1. General Idea of the Production Scheduling Algorithm for the Second Stage. Compared with that of the first-stage scheduling algorithm, the design of the second-stage scheduling algorithm is more complex because the types of tasks in the second stage are different: there are tasks both with and without waiting time. Thus, the assembly sequence must be determined for tasks both with and without waiting time. According to the set optimization objectives, the general ideas for the second stage scheduling algorithm are as follows.

First, we need to identify and distinguish the types of tasks.

Second, the tasks without waiting time are sorted in descending order of assembly time.

Third, for tasks requiring waiting time, the waiting time is defined as the time consumed by the corresponding subsystem in the first stage. Therefore, the scheduling of tasks requiring waiting time is a dynamic process that should be combined with the scheduling situation of the first stage in Section 4.1.1 and should be changed along with changes in the first stage of scheduling. Additionally, once a task is being assembled, it cannot be interrupted, and the assembly of another task can be started only if the previous task is completed.

Finally, in the second stage, the goal is to balance the workload of each assembly workstation as much as possible. Only after each assembly workstation has completed the assembly task can the product be transferred to the next stage.

4.2.2. Steps of S2

Step 1: all the assembly tasks in the second stage are classified according to their type to form the no waiting time task set K_1 and the waiting time task set K_2.

Step 2: the tasks in task set K_1 are sorted in descending order of assembly time P_k, namely, $P_k \geq P_{k+1}$, $\forall k \in \{1, \ldots, |K_1| - 1\}$.

Step 3: the first $\min\{|K_1|, |L|\}$ tasks are arranged on $\min\{|K_1|, |L|\}$ workstations in turn; now, let $k = \min\{|K_1|, |L|\} + 1, \forall k \in K_1$.

Step 4: confirm whether there is an idle assembly workstation; if so, go to Step 5. Otherwise, continue to wait until an idle workstation appears; then, go to Step 5.

Step 5: search task set K_2 to confirm whether there are any tasks to be assembled. If there are tasks to be assembled, assembly task k' with the longest assembly time is assigned to the idle assembly workstation; then, $k' = k' + 1$. Otherwise, the k-th task in K_1 is assigned to the idle assembly workstation.

Step 6: if $k > |K_1|$, go to Step 7; otherwise, let $k = k + 1$, and go to Step 4.

Step 7: confirm whether there is an idle assembly workstation; if so, go to Step 8. Otherwise, continue to wait until an idle workstation appears; then, go to Step 8.

Step 8: search task set K_2 to confirm whether there are any tasks to be assembled. If there are tasks to be assembled, assembly task k' with the longest assembly time is assigned to the idle assembly workstation; then, $k' = k' + 1$. Otherwise, continue to wait until a task to be assembled appears and assign it to the idle workstation.

Step 9: if $k' > |K_2|$, go to Step 10; otherwise, go to Step 7.

Step 10: end.

The time complexity of algorithm S2 is $O(m'n'^2)$, where m' represents the number of assembly workstations, that is, $m' = |L|$, and n' represents the maximum number of the two types of assembly tasks, that is, $|L|n' = \max\{|K_1|, |K_2|\}$. Additionally, m steps are required to confirm the existence of an idle assembly workstation in Step 4. Step 5 needs $|K|$ times to search set $|K|$. If there exist tasks to be assembled, the assignment of the task with the longest assembly time takes no more than $|K|$ iterations; otherwise, it will take no more than $|K_1|$ iterations to assign the task from set K_1 in Step 5 and Step 6. Therefore, the complexity of Step 4, Step 5, and Step 6 is $O(m'n'^2)$. Similarly, the complexity of Step 7, Step 8, and Step 9 is $O(m'|K|^2)$. Therefore, the total complexity of algorithm S1 is $O(m'n'^2) + O(m'|K|^2) = O(m'n'^2)$.

4.3. Improved Algorithm for the First Stage (S1′). The scheduling algorithm design of the second stage will change dynamically with the scheduling of the first stage. Here, we give priority to tasks with longer assembly times, which need to be assembled first, while in the scheduling algorithm design of the first stage, we give priority to subsystems with a longer processing time, which need to be processed first. However, the assembly time and processing time are not the same concept; if the scheduling algorithm of the first stage is directly embedded in the scheduling algorithm of the second stage, it is no longer applicable. Therefore, the improved scheduling algorithm of the first stage (S1′) should be adjusted as follows.

Step 1: subsystems I_k are sorted in descending order of assembly time P_k, namely, $P_k \geq P_{k+1}, \forall k \in \{1, \ldots, |K| - 1\}$

Step 2: all intermediate products in set I_1 set are sorted in descending order of t_i, namely, $t_i \geq t_{i+1}$, $\forall i \in \{1, \ldots, |I_1| - 1\}$

Step 3: let $k = 1$; the first $|J|$ components are arranged on $|J|$ workstations in turn, and now let $i = |J| + 1$

Step 4: if any workstation finishes processing, the i-th intermediate product in set I_k is assigned to this workstation

Step 5: if $i > |I_k|$, then let $k = k + 1$, and go to Step 6; otherwise, go to Step 4 and let $i = i + 1$

Step 6: if $k < |K|$, then all intermediate products in set I_k are sorted in descending order of t_i, namely, $t_i \geq t_{i+1}$; let $i = 1$, and go to Step 4; otherwise, go to Step 7

Step 7: end

The only difference between S1 and S1′ is the sorting criterion in Step 1; therefore, the complexity of S1′ is the same as that of S1.

In summary, to solve the two-stage production scheduling problem, we propose two combined scheduling algorithms in this chapter: the S1-S2 algorithm and the S1′-S2 algorithm.

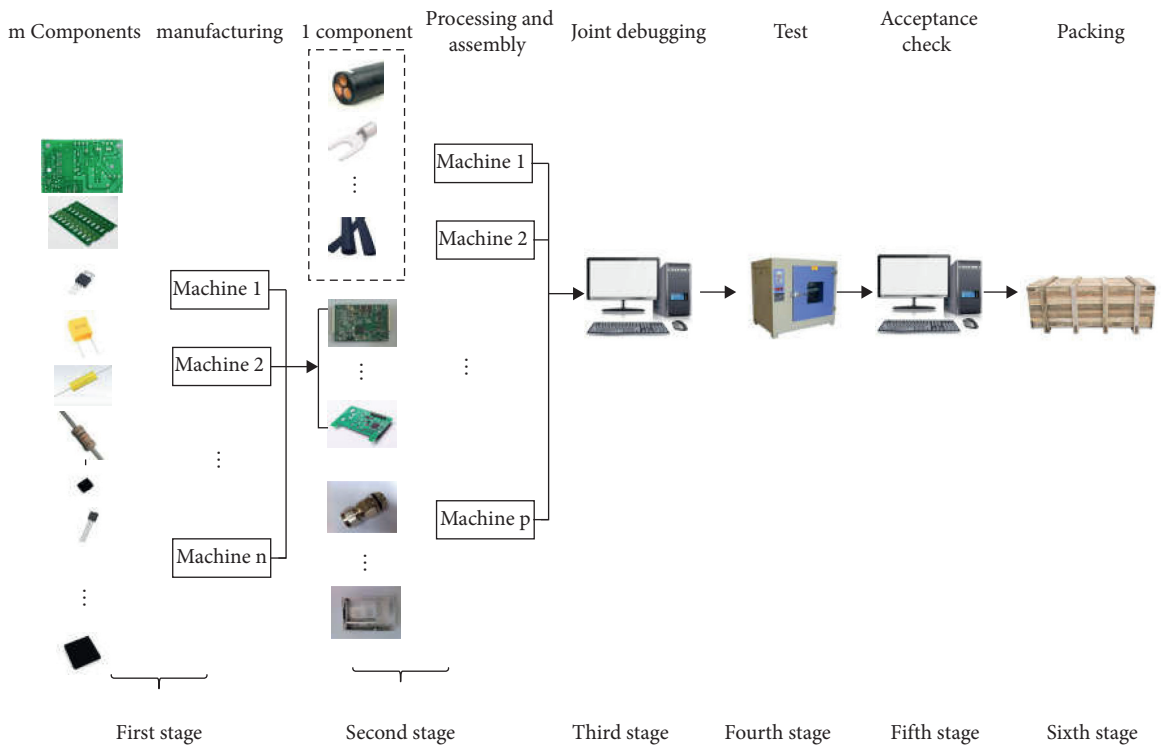

Figure 2: Schematic representation of the multistage job shop problem.

5. Numerical Experiments

Empirical enterprises produce three main types of products (type *A*, type *E*, and type *F*), and all products have to go through multistage production successively. Figure 2 illustrates a schematic of the multistage job shop problem. In the first stage, an intermediate product circuit board is produced, and, in the second stage, cable processing, core component pretreatment, and subsystem assembly are conducted. Then, joint debugging, testing, acceptance checking, and packing are performed in the subsequent stage. The entire production process conforms to the mathematical model and algorithm design built in this paper.

In the enterprise, there are four parallel processing workstations in the first stage. The four processing workstations have no sequence when they perform processing tasks, and they are independent of each other. Once an intermediate product begins processing on one of the workstations, it will not be terminated and transferred to another workstation. Furthermore, in the second stage, there are five assembly workstations. The five assembly workstations have no sequence when they perform assembly tasks, and the workstations are independent of each other. Once an assembly task is begun on a workstation, it will not be terminated and transferred to another workstation.

5.1. Process Data. An F-type product with complex technology is selected for empirical research. For reasons of confidentiality, the actual names of the subsystems, intermediate products, and assembly tasks are replaced in the following process data.

5.1.1. Process Data in the First Stage. The processing time and subsystem details of the first-stage intermediate products are shown in Table 1.

5.1.2. Process Data in the Second Stage. The assembly time and task type of the second-stage assembly tasks are shown in Table 2.

5.1.3. Process Data in the Subsequent Stage. After completing the assembly tasks in the second stage, the products are integrated into a set of system products; then, joint debugging, working environment testing, acceptance checking, and packing are conducted in the subsequent stages. Table 3 shows the process data for the subsequent stages.

5.2. Comparison Analysis. To verify the effectiveness of the S1′-S2 algorithm, we propose two benchmark approaches: the S1-S2 algorithm and the actual operation approach of the enterprise.

5.2.1. Comparative Analysis between the S1′-S2 Algorithm and S1-S2 Algorithm. The operation results of the S1′-S2 algorithm and the S1-S2 algorithm in the first and second stages, which can be seen in Tables 4 and 5, can be obtained using the scheduling algorithm in Section 4. The comparison results are shown in Table 6.

Table 1: Process data in the first stage.

Subsystem name	Intermediate product name i	Processing time t_i
Subsystem 1	Intermediate product 1	50
	Intermediate product 2	20
	Intermediate product 3	4
	Intermediate product 4	24
	Intermediate product 5	4
	Intermediate product 6	12
	Intermediate product 7	18
	Intermediate product 8	28
	Intermediate product 9	35
	Intermediate product 10	16
	Intermediate product 11	42
	Intermediate product 12	32
	Intermediate product 13	9
	Intermediate product 14	4.5
	Intermediate product 15	4
	Intermediate product 16	32
	Intermediate product 17	4
Subsystem 2	Intermediate product 18	18
	Intermediate product 19	3
	Intermediate product 20	16
	Intermediate product 21	16
	Intermediate product 22	1
	Intermediate product 23	1
Subsystem 3	Intermediate product 24	4
	Intermediate product 25	1
	Intermediate product 26	1
Subsystem 4	Intermediate product 27	10
Subsystem 5	Intermediate product 28	12
Subsystem 6	Intermediate product 29	8
	Intermediate product 30	8
	Intermediate product 31	4
	Intermediate product 32	6
	Intermediate product 33	6
	Intermediate product 34	12
Subsystem 7	Intermediate product 35	0.5

Table 2: Process data in the second stage.

Task type	Task name	Assembly time
Tasks without waiting time	Task 1	22
	Task 2	27
	Task 3	2
Tasks with waiting time	Task 4	19
	Task 5	25.5
	Task 6	8.5
	Task 7	72
	Task 8	13
	Task 9	17
	Task 10	13.5

Table 3: Process data in the subsequent stage.

Joint debugging	Working environment test	Acceptance check	Packing
4	56	3	1

Table 6 shows that 255 working hours are required to complete a single set of F-type products using the S1-S2 algorithm, while 199.5 working hours are required by the S1′-S2 algorithm: the completion time of the S1′-S2 algorithm is 21.76% higher than that of the S1-S2 algorithm.

In terms of the annual output of F-type products, the S1′-S2 algorithm produces 3.59 sets more than the S1-S2 algorithm, and the production efficiency is increased by 24.90%. The annual output of F-type products is obtained as the ratio of the actual working days of the enterprise in the whole year to the production cycle of a single set of F-type products. Two points need to be explained.

First, the working system of the empirical enterprise is a one-day-off system; that is, there is only one day off (Sunday) every week. Based on 52 weeks in a year and deducting 11 days for China national legal holidays, we can obtain the total actual number of working days for the whole year of the empirical enterprise as follows: actual working days = 365 − 11 − 52 = 302 (days).

Second, the working environment testing in the subsequent stages, which costs 56 hours, is completed continuously, so the working environment test actually takes 3 days.

In terms of the annual profit value of F-type products, the S1′-S2 algorithm yields 1.436 million Yuan greater profit than the S1-S2 algorithm, and the profit margin is increased by 24.90%.

The reason for the above differences is the different scheduling rules adopted in the production scheduling process. The S1-S2 algorithm isolates the two-stage production scheduling into two separate production stages, and its scheduling algorithm guarantees only the minimum of the latest completion time of each stage. The S1′-S2 algorithm regards the two-stage production scheduling as a whole and forms a link between the stages. Table 4 shows that the latest completion time of the first stage of the S1′-S2 algorithm is 122 working hours and that of the S1-S2 algorithm is 120 working hours. However, in the second stage, the S1′-S2 algorithm gives priority to the assembly time and returns the information of the subsystems with long assembly times to the first stage of the production scheduling, which effectively balances the assembly capacity of each assembly workstation in the second stage. Thus, a more satisfactory solution can be obtained.

5.2.2. Comparative Analysis between the S1′-S2 Algorithm and the Actual Operation of the Enterprise. To verify the effectiveness of the S1′-S2 algorithm, we obtain the output of F-type products from 2015 to 2018. The comparison results between the actual operation of the enterprise and the S1′-S2 algorithm are provided in Table 7.

In Table 7, the output of F-type products is the highest in 2018, reaching 9 sets. The S1′-S2 algorithm can be used to obtain 14.42 sets of annual production, which is 1.6 times the actual production in 2018. Therefore, the production capacity of F-type products is improved considerably. In view of the above situation, we analyze the following two reasons for this improvement.

Table 4: Comparison table of algorithm operation results in the first stage.

Workstation no.	S1′-S2 algorithm			S1-S2 algorithm		
	Intermediate product no.	Processing time t_{ij}	Processing end time C_{ij}	Intermediate product no.	Processing time t_{ij}	Processing end time C_{ij}
Processing workstation 1	Intermediate product 27	10	10	Intermediate product 1	50	50
	Intermediate product 19	3	13	Intermediate product 2	20	70
	Intermediate product 22	1	14	Intermediate product 13	9	79
	Intermediate product 23	1	15	Intermediate product 3	4	83
	Intermediate product 1	50	65	Intermediate product 18	18	101
	Intermediate product 2	20	85	Intermediate product 30	8	109
	Intermediate product 13	9	94			
	Intermediate product 14	4.5	98.5	Intermediate product 27	10	119
	Intermediate product 34	12	110.5			
	Intermediate product 28	12	122.5			
Processing workstation 2	Intermediate product 18	18	18	Intermediate product 11	42	42
	Intermediate product 12	32	50	Intermediate product 4	24	66
	Intermediate product 16	32	82	Intermediate product 6	12	78
	Intermediate product 10	16	98	Intermediate product 14	4.5	82.5
	Intermediate product 15	4	102	Intermediate product 17	4	86.5
	Intermediate product 30	8	110	Intermediate product 19	3	89.5
	Intermediate product 35	0.5	110.5	Intermediate product 22	1	90.5
				Intermediate product 23	1	91.5
	Intermediate product 24	4	114.5	Intermediate product 34	12	103.5
				Intermediate product 33	6	109.5
				Intermediate product 24	4	113.5
Processing workstation 3	Intermediate product 20	16	16	Intermediate product 9	35	35
	Intermediate product 11	42	58	Intermediate product 8	28	63
	Intermediate product 4	24	82	Intermediate product 7	18	81
	Intermediate product 6	12	94	Intermediate product 15	4	85
	Intermediate product 3	4	98	Intermediate product 21	16	101
	Intermediate product 17	4	102	Intermediate product 32	6	107
	Intermediate product 32	6	108	Intermediate product 31	4	111
	Intermediate product 33	6	114	Intermediate product 25	1	112
	Intermediate product 26	1	115	Intermediate product 26	1	113
				Intermediate product 35	0.5	113.5
Processing workstation 4	Intermediate product 21	16	16	Intermediate product 12	32	32

Table 4: Continued.

Workstation no.	S1′-S2 algorithm			S1-S2 algorithm		
	Intermediate product no.	Processing time t_{ij}	Processing end time C_{ij}	Intermediate product no.	Processing time t_{ij}	Processing end time C_{ij}
	Intermediate product 9	35	51	Intermediate product 16	32	64
	Intermediate product 8	28	79	Intermediate product 10	16	80
	Intermediate product 7	18	97	Intermediate product 5	4	84
	Intermediate product 5	4	101	Intermediate product 20	16	100
	Intermediate product 29	8	109	Intermediate product 29	8	108
	Intermediate product 31	4	113	Intermediate product 28	12	120
	Intermediate product 25	1	114			

Table 5: Comparison table of algorithm operation results in the second stage.

Workstation no.	S1′-S2 algorithm				S1-S2 algorithm			
	Task no.	Assembly time P_k	Waiting time	Assembly end time C_{kl}	Task no.	Assembly time P_k	Waiting time	Assembly end time C_{kl}
Assembly workstation 1	Task 2	27	0	27	Task 2	27	0	27
	Task 9	17	114	131	Task 10	13.5	113.5	127
Assembly workstation 2	Task 1	22	0	22	Task 1	22	0	22
	Task 10	13.5	110.5	124	Task 6	8.5	113.5	122
					Task 8	13	120	135
Assembly workstation 3	Task 3	2	0	2	Task 3	2	0	2
	Task 4	19	102	121	Task 9	17	111	128
Assembly workstation 4	Task 7	72	10	82	Task 4	19	86.5	105.5
	Task 8	13	122.5	135.5	Task 7	72	119	191
Assembly workstation 5	Task 5	25.5	18	43.5	Task 5	25.5	101	126.5
	Task 6	8.5	115	123.5				

Table 6: Comparison of the effectiveness of the algorithms.

	S1-S2 algorithm	S1′-S2 algorithm	Effectiveness comparison (%)
Total completion time of single set of *F*-type products (hours)	255	199.5	21.76
Annual output of *F*-type product (set)	10.83	14.42	24.90
Annual profit value of *F*-type product (10000 yuan)	433.2	576.8	24.90

Table 7: Production of *F*-type products in 2015–2018.

	Annual output of *F*-type product				S1′-S2 algorithm
Year	2015	2016	2017	2018	14.42
Production (set)	5	5	7	9	

First, in the actual production scheduling process, enterprises usually implement production scheduling according to the production schedule. The production scheduling personnel not only are familiar with the production process of the product but also monitor the first-line production workshop to follow the production progress and take timely and effective adjustment measures to ensure that the production scheduling meets the requirements of the production schedule as much as possible. However, in the process of adjusting production scheduling, due to the different experience of production scheduling personnel, different scheduling rules will be adopted, which results in the balance of orderly production of workstations being broken. In the end, good production capacity cannot be obtained.

Second, the empirical enterprise implements the time work system. Under this condition, production workers inevitably experience inertia that affects their work enthusiasm and initiative and leads to relatively low production efficiency. The S1′-S2 algorithm that we highly recommend is designed from the theoretical perspective. It eliminates the influence of human factors in the

production scheduling process to improve production capacity.

In summary, the effectiveness of the S1′-S2 algorithm is substantially better than that of the S1-S2 algorithm and the actual operation of enterprises, which demonstrates that the empirical enterprise can effectively improve its production capacity and earn greater profits by introducing the S1′-S2 algorithm. The S1′-S2 algorithm has good reference significance for guiding the actual production scheduling of enterprises.

6. Conclusions

This paper constructs a multistage production scheduling model and designs a multistage production scheduling algorithm based on sequencing theory by analyzing the relationships among production stages and optimizing each stage. Meanwhile, to verify the effectiveness of the construction model and design algorithm, we conduct numerical experiments that show that the model and algorithm constructed in this paper not only improve the production efficiency of an empirical enterprise but also provide technical support to improve production capacity.

However, the paper considers only static scheduling and not dynamic scheduling. In the actual production scheduling process, it is impossible for the production personnel to work all the time. Moreover, production equipment cannot operate all the time without faults, and not all the raw materials can be delivered on time. Therefore, there are many uncertain factors in the production scheduling process. Thus, one direction of follow-up research will be to improve the adaptability of the production scheduling method.

Data Availability

The data used to support the findings of this study can be found in Tables 1–3 of this paper.

Conflicts of Interest

The authors declare no conflicts of interest.

Authors' Contributions

Conceptualization was performed by Jianzhong Xu and Song Zhang; formal analysis was done by Song Zhang and Yuzhen Hu; investigation was carried out by Song Zhang; methodology was prepared by Song Zhang and Yuzhen Hu; original draft was written by Song Zhang; review and editing were carried out by Song Zhang; supervision was performed by Jianzhong Xu. All authors have read and approved the published version of the manuscript.

Acknowledgments

This research was partially supported by the Natural Science Foundation of Heilongjiang Province (LH2019G014 and QC2016095), Philosophy and Social Science Research Planning Project of Heilongjiang Province (17JYH49 and 18GLC208), Key Research Topics of Economic and Social Development in Heilongjiang Province (19023), the 2019 Annual Basic Project of the Party's Political Construction Research Center of the Ministry of Industry and Information Technology (19GZY411), the National Natural Science Foundation on Emergency Management Project of China (71841054), the National Natural Science Foundation of China (71801061), the project of Central University Basic Research Fund (3072020CFT0902), and Philosophy and Social Science Research Planning Project of Zhejiang Province (20NDQN297YB).

References

[1] X. Shao and Y. Rao, *Theory and Method of Manufacturing System Operation Optimization*, Science Press, Beijing, China, 2010.

[2] S. M. Johnson, "Optimal two- and three-stage production schedules with setup times included," *Naval Research Logistics Quarterly*, vol. 1, no. 1, pp. 61–68, 1954.

[3] S. C. Graves, "A review of production scheduling," *Operations Research*, vol. 29, no. 4, pp. 646–675, 1981.

[4] J. R. Jackson, "Scheduling a production line to minimize maximum tardiness," Research Report 43, UCLA, Los Angeles, CA, USA, 1955.

[5] W. E. Smith, "Various optimizers for single-stage production," *Naval Research Logistics Quarterly*, vol. 3, no. 1-2, pp. 59–66, 1956.

[6] T. C. Hu, "Parallel sequencing and assembly line problems," *Operations Research*, vol. 9, no. 6, pp. 841–848, 1961.

[7] R. L. Graham, "Bounds for certain multiprocessing anomalies," *Bell System Technical Journal*, vol. 45, no. 9, pp. 1563–1581, 1966.

[8] R. L. Graham, "Bounds on multiprocessing timing anomalies," *SIAM Journal on Applied Mathematics*, vol. 17, no. 2, pp. 416–429, 1969.

[9] Z. Jiang, F. Chen, and X. Zhang, "Single-machine scheduling with times-based and job-dependent learning effect," *Journal of the Operational Research Society*, vol. 68, no. 7, pp. 809–815, 2017.

[10] K. Kianfar, G. Moslehi, and A. S. Nookabadi, "Exact and heuristic algorithms for minimizing Tardy/Lost penalties on a single-machine scheduling problem," *Computational and Applied Mathematics*, vol. 37, no. 2, pp. 867–895, 2018.

[11] X. Cao, W.-H. Wu, W.-H. Wu, and C.-C. Wu, "Some two-agent single-machine scheduling problems to minimize minmax and minsum of completion times," *Operational Research*, vol. 18, no. 2, pp. 293–314, 2018.

[12] Z. Xu and D. Xu, "Single-machine scheduling with workload-dependent tool change durations and equal processing time jobs to minimize total completion time," *Journal of Scheduling*, vol. 21, no. 4, pp. 461–482, 2018.

[13] A. Ekici, M. Elyasi, O. Ö. Özener, and M. B. Sarıkaya, "An application of unrelated parallel machine scheduling with sequence-dependent setups at vestel electronics," *Computers & Operations Research*, vol. 111, pp. 130–140, 2019.

[14] J. C. Yepes-Borrero, F. Villa, F. Perea, and J. P. Caballero-Villalobos, "GRASP algorithm for the unrelated parallel machine scheduling problem with setup times and additional resources," *Expert Systemswith Applications*, vol. 141, Article ID 112959, 2020.

[15] M. S. S. Mir, J. Rezaeian, and H. Mohamadian, "Scheduling parallel machine problem under general effects of deterioration and learning with past-sequence-dependent setup time:

heuristic and meta-heuristic approaches," *Soft Computing*, vol. 24, no. 2, pp. 1335–1355, 2020.

[16] W. Luo, Y. Xu, W. Tong, and G. Lin, "Single-machine scheduling with job-dependent machine deterioration," *Journal of Scheduling*, vol. 22, no. 6, pp. 691–707, 2019.

[17] R. Chen and J. Yuan, "Unary NP-hardness of single-machine scheduling to minimize the total tardiness with deadlines," *Journal of Scheduling*, vol. 22, no. 5, pp. 595–601, 2019.

[18] W. Luo and F. Liu, "On single-machine scheduling with workload-dependent maintenance duration," *Omega*, vol. 68, pp. 119–122, 2017.

[19] S. Atakan, B. Kerem, and N. Noyan, "Minimizing value-at-risk in single-machine scheduling," *Annals of Operations Research*, vol. 248, no. 1-2, pp. 25–73, 2017.

[20] L. Zhang, Q. Deng, G. Gong, and W. Han, "A new unrelated parallel machine scheduling problem with tool changes to minimise the total energy consumption," *International Journalof Production Research*, pp. 1–20, 2019.

[21] E. Vallada, F. Villa, and L. Fanjul-Peyro, "Enriched meta-heuristics for the resource constrained unrelated parallel machine scheduling problem," *Computers & Operations Research*, vol. 111, pp. 415–424, 2019.

[22] B. Alidaee, H. Wang, R. B. Kethley, and F. Landram, "A unified view of parallel machine scheduling with interdependent processing rates," *Journal of Scheduling*, vol. 22, no. 5, pp. 499–515, 2019.

[23] B. Vincent, C. Duhamel, L. Ren, and N. Tchernev, "A population-based metaheuristic for the capacitated lot-sizing problem with unrelated parallel machines," *International Journal of Production Research*, pp. 1–18, 2020.

[24] F. M. B. Toledo and V. A. Armentano, "A Lagrangian-based heuristic for the capacitated lot-sizing problem in parallel machines," *European Journal of Operational Research*, vol. 175, no. 2, pp. 1070–1083, 2006.

[25] N. Ueno, S. Sotojima, and J. Takeda, "Multi-stage flow-shop in steel works," in *Proceedings of the 24th Annual Simulation Symposium*, IEEE Computer Society Press, New Orleans, LO, USA, pp. 332–337, 1991.

[26] S. Yuan, T. Li, B. Wang, and N. Yu, "Model and algorithm for two-stage flow shop group scheduling problem with special blocking constraint," *Control and Decision*, vol. 35, no. 7, pp. 1773–1779, 2020.

[27] W. Qin, Z. Zhuang, Y. Liu, and O. Tang, "A two-stage ant colony algorithm for hybrid flow shop scheduling with lot sizing and calendar constraints in printed circuit board assembly," *Computers& Industrial Engineering*, vol. 138, Article ID 106115, 2019.

[28] M. K. Marichelvam, M. Geetha, and O. Tosun, "An improved particle swarm optimization algorithm to solve hybrid flowshop scheduling problems with the effect of human factors—a case study," *Computers& Operations Research*, vol. 114, Article ID 104812, 2020.

[29] S. Li, Y. Li, and Z. Liu, "Mix optimization scheduling approach formulti-resource job-shop," *Journal of Systems Engineering*, vol. 22, no. 5, pp. 551–555, 2007.

[30] S. Wang, Y. Li, and W. Huai, "A three-stage decoding method for just-in-time job-shop scheduling," *Industrial Engineering Journal*, vol. 18, no. 3, pp. 98–104, 2015.

[31] P. Fattahi, N. B. Rad, F. Daneshamooz, and S. Ahmadi, "A new hybrid particle swarm optimization and parallel variable neighborhood search algorithm for flexible job shop scheduling with assembly process," *Assembly Automation*, vol. 40, no. 3, p. 419, 2020.

[32] W. Huang, C. Yan, X. Wang, and J. Xu, "Job shop scheduling problem for complex engineering project products," *Mechanical Science and Technology for Aerospace Engineering*, vol. 39, no. 3, pp. 1–12, 2020.

[33] B.-h. Zhou, X.-m. Liao, and K. Wang, "Kalman filter and multi-stage learning-based hybrid differential evolution algorithm with particle swarm for a two-stage flow shops scheduling problem," *Soft Computing*, vol. 23, no. 24, pp. 13067–13083, 2019.

[34] M. Liu, X. Yao, and Y. Li, "Hybrid whale optimization algorithm enhanced with Levy flight and differential evolution for job shop scheduling problems," *Applied Soft Computing*, vol. 87, Article ID 105954, 2020.

[35] D. Ferone, S. Hatami, E. M. Gonzalez-Neira, A. A. Juan, and P. Festa, "A biased-randomized iterated local search for the distributed assembly permutation flow-shop problem," *International Transactionsin Operational Research*, vol. 27, no. 3, 2020.

[36] S. Liu, J. Pei, H. Cheng, X. Liu, and P. M. Pardalos, "Two-stage hybrid flow shop scheduling on parallel batching machines considering a job-dependent deteriorating effect and non-identical job sizes," *Applied Soft Computing*, vol. 84, 2019.

[37] H. Samarghandi, "Minimizing the makespan in a flow shop environment under minimum and maximum time-lag constraints," *Computers & Industrial Engineering*, vol. 136, pp. 614–634, 2019.

[38] G. Gong, R. Chiong, Q. Deng et al., "Energy-efficient flexible flow shop scheduling with worker flexibility," *Expert Systemswith Applications*, vol. 141, Article ID 112902, 2020.

[39] R. H. Caldeira and A. Gnanavelbabu, "Solving the flexible job shop scheduling problem using an improved Jaya algorithm," *Computers& Industrial Engineering*, vol. 137, Article ID 106064, 2019.

[40] A. Dabah, A. Bendjoudi, A. AitZai, and N. N. Taboudjemat, "Efficient parallel tabu search for the blocking job shop scheduling problem," *Soft Computing*, vol. 23, no. 24, pp. 13283–13295, 2019.

[41] Y. Zhang, H. Zhu, and D. Tang, "An improved hybrid particle swarm optimization for multi-objective flexible job-shop scheduling problem," *Kybernetes*, 2019, in Press.

[42] H. Tang and C. Zhao, *Introduction to Sequencing Theory*, pp. 45–59, Science Press, Beijing, China, 2002.