symmetry

MDPI

Article

An Improved Multi-Objective Evolutionary Approach for Aerospace Shell Production Scheduling Problem

Qing Wang [1], Xiaoshuang Wang [2], Haiwei Luo [3] and Jian Xiong [4,*]

1 College of Systems Engineering, National University of Defense Technology, Changsha 410073, China; wangqing_1970@126.com or wq158@sina.com
2 College of Information Communication, National University of Defense Technology, Wuhan 430014, China; frank1188@163.com
3 Capital Aerospace Machinery Corporation Limited, Beijing 100044, China; luohaiwei85@163.com
4 School of Business Administration, Southwestern University of Finance and Economics, Chengdu 610074, China
* Correspondence: xiongjian2017@swufe.edu.cn

Received: 19 February 2020; Accepted: 12 March 2020; Published: 1 April 2020

Abstract: To certain degree, multi-objective optimization problems obey the law of symmetry, for instance, the minimum of one objective function corresponds to the maximum of another objective. To provide effective support for the multi-objective operation of the aerospace product shell production line, this paper studies multi-objective aerospace shell production scheduling problems. Firstly, a multi-objective optimization model for the production scheduling of aerospace product shell production lines is established. In the presented model, the maximum completion time and the cost of production line construction are optimized simultaneously. Secondly, to tackle the characteristics of discreteness, non-convexity and strong NP difficulty of the multi-objective problem, a knowledge-driven multi-objective evolutionary algorithm is designed to solve the problem. In the proposed approach, structural features of the scheduling plan are extracted during the optimization process and used to guide the subsequent optimization process. Finally, a set of test instances is generated to illustrate the addressed problem and test the proposed approach. The experimental results show that the knowledge-driven multi-objective evolutionary algorithm designed in this paper has better performance than the two classic multi-objective optimization methods.

Keywords: aerospace shell production scheduling problem; multi-objective optimization; evolutionary algorithm; knowledge-driven optimization

1. Introduction

Integral shell parts are the core components of aerospace products, which have complex feature structures and high machining accuracy requirements. With the development of high-density aerospace products, the tasks of aerospace shell products gradually show the characteristics of multiple models, multiple states, frequent changes, short development cycles, heavy tasks, high quality and reliability requirements, and the coexistence of development and batch production. Digital and intelligent manufacturing methods change the manufacturing model. In a production line, several types of shell products need to be processed given specific constraints and objectives. An effective production plan plays an important role for the digital production of aerospace shell products.

The aerospace shell production scheduling problem shares the characteristics of production scheduling problems and job shop scheduling problems. Due to the importance in practice, researchers have conducted various studies in this area. In [1], the researchers addressed a type of flexible job shop scheduling problem where the costs of job earliness, tardiness, and operations were minimized.

A multistage graph-based heuristic algorithm was developed to solve the problem. The distributed assembly flexible job shop scheduling problem (DAFJSP) was studied in [2], where an improved differential evolution simulated annealing algorithm was proposed to solve the problem. In [3], a single goal flexible JSSP (FJSSP) was solved by a social spider optimization (SSO). In [4], a flexible job shop scheduling with interval grey processing time was considered and elitism genetic algorithm with elitism strategy in external memory was proposed as the solution technique. A hybrid sequential genetic algorithm was designed to solve a flexible job shop scheduling problem with lot streaming in [5]. In [6], the authors developed a teaching–learning-based algorithm to solve the FJSP with fuzzy processing time. In [7], a similar problem with fuzzy duration was addressed and a bionic artificial bee colony algorithm was used to solve the problem. A remanufacturing process with fuzzy processing time was considered in [8] and a discrete harmony search algorithm was presented as the solution technique. In [9], the authors employed a list-based search algorithm to solve the FJSP with sequencing flexibility. In [10], a simulation-based optimization approach was employed to solve a dynamic FJSP. In [11], the researchers studied the multi-process production scheduling problem, considered the optimization of renewable energy maximization, and adopted a two-stage robust optimization to solve the problem of uncertainty. In [12], the researchers addressed the production scheduling problem considering the status of the machine and equipment, and proposed a model combining the monitoring of equipment status and production planning and scheduling. In [13], the researchers studied how to improve manufacturing efficiency through production scheduling in an agent-based supply chain system. In addition, a large number of scholars have carried out applied research on production scheduling in various fields, such as the mining industry [14–16], the steel industry [17], the pharmaceutical industry [18], the chemical industry [19], and other complex production systems [20].

Although there are a lot works addressing job shop scheduling problems, relatively few studies on aerospace production lines have been reported. Furthermore, most current relative research focuses on single-objective optimization problems. Specifically, the completion time of tasks, i.e., the makespan, is the main objective to be optimized. The aerospace product shell production line is a complex production system. The operation of complex systems requires comprehensive consideration of multiple indicators to achieve a balance of efficiency, economy and environment. Generally speaking, efficiency indicators and economic indicators are conflicting. After considering these two objectives in the optimization process of the aerospace product shell production line, its optimization problem is transformed into a multi-objective optimization problem, and a series of feasible solutions to be weighed need to be provided for decision-makers. Thus, in this paper, we address the multi-objective aerospace shell production scheduling problem (MOASPSP). The existing approaches mentioned in the above literature were designed for single-objective job shop scheduling problems. These algorithms are not suitable for the MOASPSP addressed in our paper. To the best of our knowledge, the study presented in this paper is the first work about a multi-objective model for aerospace shell production scheduling. Since there is no existing algorithm for solving the problem, a multi-objective approach is designed and tested for the addressed MOASPSP.

The rest of this paper is organized as follows. In Section 2, the MOASPSP is described and formulated. To solve the problem, a knowledge-driven multi-objective evolutionary algorithm is developed and presented in Section 3. Experimental analysis is conducted in Section 4. Finally, we summarize the conclusions of this study and provide a future research direction.

2. Multi-Objective Aerospace Shell Production Scheduling Problem

2.1. General Problem Description

An aerospace shell production scheduling problem (ASPSP) can be described as follows: In an aerospace shell product production line, there are several types of machines. Each type of machine has a specific amount. The workshop's production task consists of a set of independent jobs. Each job has a sequence of operations. Each operation can be processed by a specific type of machine. To achieve certain targets, all operations should be properly assigned to available machines. Furthermore, the sequence of operation processing needs to be decided.

2.2. Mathematical Formulation

In this subsection, mathematical formulation of the addressed problem is presented. To describe the problem conveniently, we use notations as presented in Table 1.

Table 1. Notations for problem formulation.

Symbol	Description
n	Number of jobs.
e	Number of machine types.
i	Index of jobs, $i = 1, 2, \ldots, n$
j	Index of machine types, $j = 1, 2, \ldots$, e
a_j	Machine amount of type j
j,m	Index of machine for type j, $m = 1, 2, \ldots, a_j$
C_j	The unit cost of machine type j.
$machine_{j,m}$	The mth machine for the type j.
J	The set of jobs, $J = \{J_1, J_2, \ldots, J_n\}$
F_i	The finish time of job i.
q_i	The number of operations for job i.
k	The index of operation for job i.
O_i	The set of operation for job i
O	The set of all operations, $O = \{O_1, O_2, \ldots, O_n\}$
$opr_{i,k}$	The kth operation of job i.
$Mach\_type_{i,k}$	The type of machine that can process operation $opr_{i,k.}$
$d_{i,k}$	The duration of operation $opr_{i,k.}$
$st_{i,k}$	The start time of operation $opr_{i,k.}$
$et_{i,k}$	The end time of operation $opr_{i,k.}$
$x_{i,k,j,m}$	Machine assignment index of operation $opr_{i,k}$ to $machine_{j,m.}$
M	The set of machines deployed in the production line.
s	Index of machine in the set of M, $s = 1, 2, \ldots, \lvert M \rvert$
$y^t{}_{j,m,i,k}$	At time t, whether a machine $machine_{j,m}$ is processing operation $opr_{i,k}$.

In aerospace shell production lines, there are n independent jobs. The set of jobs is denoted as $J = \{J_1, J_2, \ldots, J_\mathrm{n}\}$. Each job J_i includes a sequence of operations, denoted as O_i. The number of operations for job J_i is q_i. The duration of each duration $opr_{i,k}$ is represented as $d_{i,k}$. There are t types of machine in the production line. Each type of machine consists of a_j available machines. The unit cost for machine type j is C_j. Each operation will be processed by a specific type of machine, denoted as $Mach\_type_{i,k}$.

In the problem formulation, there are several reasonable assumptions, as follows.

(1) Each operation must be processed once and only once on each machine.
(2) The same job contains a specific sequence of operations, which must be processed in the order of the operations.
(3) There are no dependencies between operations belonging different jobs.
(4) Operations scheduled to designated machines have a deterministic processing time.
(5) Each machine can only process a single operation at a specific time duration.
(6) Once processing is started, it cannot be interrupted.
(7) The situation of a machine failure is not considered.
(8) There are no differences between machines that can perform the same operation.
(9) A machine can start the processing of another operation immediately after completing one operation.

2.3. Multi-Objective Model

In practical multi-objective optimization problems, some objectives are consistent and can be converted into single objective by weighting. For instance, in our previous research, the objective of optimization is to minimize the makespan and the sum of the waiting times between operations [21]. The makespan of a schedule and the sum of the waiting time between operations are consistent to a certain extent. However, some objectives are in conflict with each other to some extent. On one hand, makespan is the main objective for an ASPSP. On the other hand, in the actual production scheduling process, the cost of production line construction needs to be considered. Intuitively, the more machine equipment deployed on the production line, the larger its production capacity, and the shorter the production completion time when completing the same batch of tasks. On the other hand, the cost of production line construction is also higher. In other words, for an ASPSP, the makespan and the cost are two conflicting objectives. Decision-makers should make a tradeoff between these objectives and select the final schedule from a set of feasible solutions. By simultaneously considering makespan and cost, an ASPSP is modeled as a multi-objective ASPSP, termed as MOASPSP.

Measures of makespan and cost are presented as follows. For each job J_i, its finish time is denoted as F_i. The makespan of a schedule indicates the maximum of finish times for all jobs and can be represented as $\max\{F_i | i = 1, 2, \ldots, n\}$. In this paper, another objective is the cost of the production line. Please note that the cost refers to fixed cost of the production line, i.e., the cost of deploying machines in the production line. We denote the set of machines in a production line as M. The number of machines in the production line is $|M|$. Then, the cost can be calculated as follows:

$$\text{Cost} = \sum_{s=1}^{|M|} C_{machine_s} | machine_s \in M \tag{1}$$

where $machine_s$ represents the sth machine in the set M, and $C_{machine_s}$ is the unit cost of $machine_s$.

Then, the mathematical model of a MOASPSP can be given as follows:

$$obj.\ \min f_1 = \max\{F_i | i = 1, 2, \ldots, n\} \tag{2}$$

$$\min f_2 = \sum_{s=1}^{|M|} C_{machine_s} | machine_s \in M \tag{3}$$

$$\text{s.t. } et_{i,k} = st_{i,k} + d_{i,k} \tag{4}$$

$$st_{i,k+1} \geq et_{i,k} \tag{5}$$

$$\sum_{j=1}^{t} \sum_{m=1}^{a_j} x_{i,k,j,m} = 1, \forall i, k \tag{6}$$

$$y^t{}_{j,m,i,k} = 1 \text{ or } y^t{}_{j,m,i,k} = 0,\ st_{i,k} \leq t \leq et_{i,k}, \forall i, k, j, m \tag{7}$$

$$y^t{}_{j,m_1,i,k} y^t{}_{j,m_2,i,k} = 0, (m_1 \neq m_2) \tag{8}$$

$$y^t{}_{j,m,i_1,k_1} y^t{}_{j,m,i_2,k_2} = 0, (i_1 \neq i_2 || k_1 \neq k_2) \tag{9}$$

In the above models, Equations (2) and (3) are two objectives which are to be minimized. Equations (4) to (9) indicate different constraints. Equation (4) indicates that an operation must be processed continuously. Equation (5) indicates that an operation only can be started after its predecessor is finished. In Equation (6), $x_{i,k,j,m}$ is an index of machine assignment, there is $x_{i,k,j,m} = 1$ if operation $opr_{i,k}$ is assigned to $machine_{j,m}$, otherwise $x_{i,k,j,m} = 0$. Then, this constraint represents that an operation can only be assigned to one machine. In Equation (7), $y^t{}_{j,m,i,k}$ is an index to show, at time t, whether a machine $machine_{j,m}$ is processing operation $opr_{i,k}$. $y^t{}_{j,m,i,k} = 1$ indicates that $machine_{j,m}$ is processing $opr_{i,k}$ at time t, otherwise $y^t{}_{j,m,i,k} = 0$. The constraint shown in Equation (7) indicates that the machine cannot be interrupted during the process. Equation (8) indicates that an operation cannot be processed on two or more machines. Equation (9) shows that a machine cannot process two or more operations at any time.

The problem presented in Equations (1) to (9) is a constrained optimization problem, which is widely used to model similar optimization problems in other areas, such as network resource allocation [21] and maximization of the overall throughput of a multiuser communication systems [22]. The addressed MOASPSP belongs to a typical type of combinatorial optimization problem, which has the characteristics of discrete and non-convex Pareto front in the objective space. The problem has a strong NP-hard characteristic. Although it is difficult to use analytical approaches to solve the problem, we are aware of that analytical approaches such as algorithm based on Lagrange multipliers are useful to analyze the bounds of a simpler problem. It is worthwhile to address this issue in our future researches.

3. An Improved Multi-Objective Evolutionary Algorithm

Traditional multi-objective optimization methods based on mathematical programming have continuous and inducible requirements on optimization goals and constraints, and are not suitable for solving combinatorial optimization problems. Therefore, this paper uses a multi-objective evolutionary algorithm (MOEA) to solve the problem.

3.1. Multi-Objective Evolutionary Algorithms

Evolutionary algorithm (EA), which is inspired by biological evolution, is a generic population-based metaheuristic optimization algorithm. An EA employs several operators such as selection, crossover and mutation to form new population. For a problem to be optimized, candidate solutions are considered as individuals in a population. The quality of the individuals is determined by fitness function. Evolution of the population then takes place after the repeated application of the above operators. For multi-objective evolutionary algorithms (MOEAs), the parent population and offspring are combined and sorted in order to generate a population for the next generation [23]. Classic MOEAs include non-dominated sorting-based approaches such as NSGA-II [24], indicator-based approaches, decomposition-based approaches such as MOEA/D [25], etc. Since NSGA-II has a remarkable performance in solving combinatorial optimization problems, in this work, we employ NSGA-II as the baseline algorithm.

3.2. Mechanism of NSGA-II

In the MOEA based on the Pareto dominance relationship, two solutions can be compared through the Pareto dominance relationship. The dominance information is used to guide the selection of the solution set. MOEAs based on Pareto dominance have always been a hot research direction, and researchers have proposed many algorithms. Among them, the NSGA-II algorithm is one of the classic algorithms based on the Pareto domination relationship [24]. In NSGA-II, individuals in the population are sorted by a Pareto relationship through a fast non-dominated sorting algorithm, and a non-dominated index is assigned to each individual. For individuals of the same non-dominant rank, the diversity of individuals is measured by the crowding distance. The overall flow chart of NSGA-II is shown in Figure 1.

3.2.1. Fast Non-Dominated Sorting Mechanism

NSGA-II uses the fast non-dominated sorting algorithm to sort the individuals of the population. The mechanism of fast non-dominated sorting is shown as in Algorithm 1. In the algorithm, P represent the population, p and q are individuals in the population. n_p indicates the number of individuals which dominate p in the population. S_p is the set of individuals which dominate individual p. $Front_i$ is the ith non-dominated front in the population.

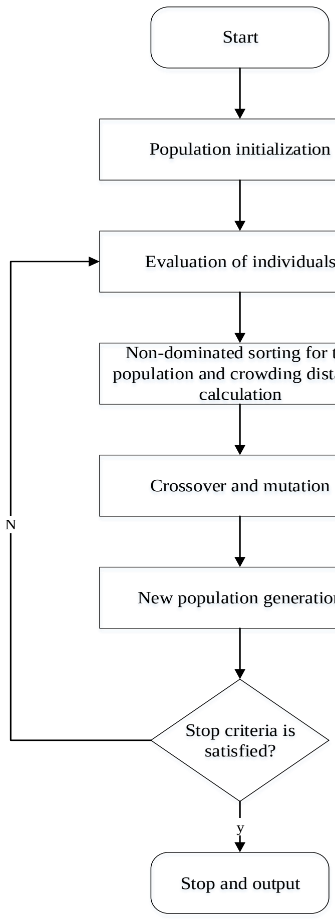

Start
Population initialization
Evaluation of individuals
Non-dominated sorting for the population and crowding distance calculation
Crossover and mutation
New population generation
Stop criteria is satisfied?
N
y
Stop and output

Figure 1. The overall workflow of NSGA-II.

Algorithm 1. Fast non-dominated sorting algorithm.

```
For each $p \in P$
$S_p = \varnothing$
$n_p = 0$
For each $q \in P$
If ($p \prec q$) then          //If $p$ dominates $q$
$S_p = S_p \cup \{q\}$     //Add $q$ to the set of solutions dominated by $p$
Else if ($q \prec p$)
$n_p = n_p + 1$        //Increment the domination counter of $p$
If $n_p = 0$ then              //$p$ belongs to the first front
$p_{rank} = 1$
$Front_1 = Front_1 \cup \{p\}$
$i = 1$                           //Initialize the front counter
While $Front_i \neq \varnothing$
$Q \neq \varnothing$                       //Used to store the members of the next front
For each $p = Front_i$
For each $q = S_p$
$n_q = n_q - 1$
If $n_q = 0$ then      //$q$ belongs to the next front
$q_{rank} = i + 1$
$Q = Q \cup \{q\}$
$i = i + 1$
$F_i = Q$
```

3.2.2. Crowding Distance Calculation

According to the fast non-dominated sorting algorithm, each individual in the population can be assigned a non-dominated rank, and the population can be divided into many non-dominated frontiers. Individuals on the same non-dominated frontier have the same value of the non-dominated rank. The lower the non-domination rank, the better. Individuals on the first non-domination frontier have a non-domination rank of 1. To further distinguish individuals on the same level of the non-dominant frontier, NSGA-II defines crowding-distance to measure individual diversity. The crowded distance is defined as the average length of the largest rectangle of the area around the individual that contains only the individual itself. The calculation of crowding distance is shown as in Algorithm 2.

Algorithm 2. Calculation of crowding distance.

```
Input $\Gamma$                                    //$\Gamma$ is the set of individuals
$l = |\Gamma|$                                  //$l$ is the number of individuals in $\Gamma$
For each $i$, set $\Gamma[i]_{dis\tan ce}$=0                //initialize distance
For each objective $m$
$\Gamma$=sort($\Gamma$,$m$)                          //sort using each objective value
$\Gamma[1]_{dis\tan ce} = \Gamma[l]_{dis\tan ce} = \infty$ //so that boundary points are always selected
For $i$=2 to ($l$-1)
$\Gamma[i]_{dis\tan ce} = \Gamma[i]_{dis\tan ce} + (\Gamma[i+1].m - \Gamma[i-1].m)\,/\,(f_m^{\max} - f_m^{\min})$
```

One can see from Algorithm 2 that individuals located on the same non-dominated frontier have the largest crowding distance on the boundary between the two ends. Thus, two boundary points will always be selected when selecting.

3.2.3. Partial Order of Individuals

For each individual in the population, there are two attributes: non-domination rank (i_{rank}) and crowding distance ($i_{dis\tan ce}$). Then, a partial order between two individuals can be defined as follows. For two individuals i and j, there is $i <_n j$ if 1) $i_{rank} < j_{rank}$; 2) $i_{rank} = j_{rank}$ and $i_{dis\tan ce} > j_{dis\tan ce}$.

After defining the partial order relationship between two individuals, any two individuals in the population can be compared. According to the definition of the partial order relationship, when two individuals belong to different ranks of a non-dominated order, the individual with the lower non-dominated level is selected. When two individuals belong to the same non-dominated rank, the individual with larger crowding distance will be selected.

3.3. Chromosome, Crossover and Mutation

The NSGA-II algorithm is aimed at multi-objective function optimization problems. However, the addressed MOASPSP is a type of combinatorial optimization problem. Therefore, individual coding needs to be redesigned according to the characteristics of multiple solving problems. In the MOASPSP, there are two types of decision variables: the order of the operations and the machine assignment of each operation. Thus, in the chromosome representation, a hybrid encoding approach is employed.

The structure of the hybrid encoding is shown in Figure 2. The chromosome consists of two parts: operation sequence and machine assignment. The length of each part of the chromosome is equal to the number of all operations, denoted as $|O|$. The gene in the chromosome of operation sequence represents an operation. The position of the operation in the chromosome indicates the processing order of the operation. In the machine assignment chromosome, the value corresponding to an operation represents its assigned machine.

The genetic operations used are crossovers and mutations. For MOASPSP, the operators of crossover and mutation remain the same as single objective ASPSP [26].

operation sequence	opr_1	opr_2	……	$opr_{\lvert O \rvert}$
machine assignment	$machine_1$	$machine_2$	……	$machine_{\lvert O \rvert}$

Figure 2. The structure of a hybrid encoding.

3.4. Knowledge-Driven MOEA

MOEAs, as a type of population-based optimization algorithm, generate a large number of individuals in each generation in the optimization process. Each individual contains data such as the structure of the solution as well as degree of strength of the solution. In the optimization process, we can continuously extract the knowledge related to the optimization problem, and then apply this knowledge to the subsequent optimization process. Based on this idea, this paper proposes a knowledge-driven multi-objective evolutionary algorithm (KD-MOEA). The workflow of the proposed KD-MOEA is shown in Figure 3. One can see from the figure that the proposed KD-MOEA includes two key operations: knowledge extraction and knowledge use. In the proposed KD-MOEA, the data contained in the population is sent to the knowledge extraction process. The data is transformed to structural knowledge, which is used to guide the subsequent process of optimization. Specifically, the obtained knowledge is used to generating new individuals for the population.

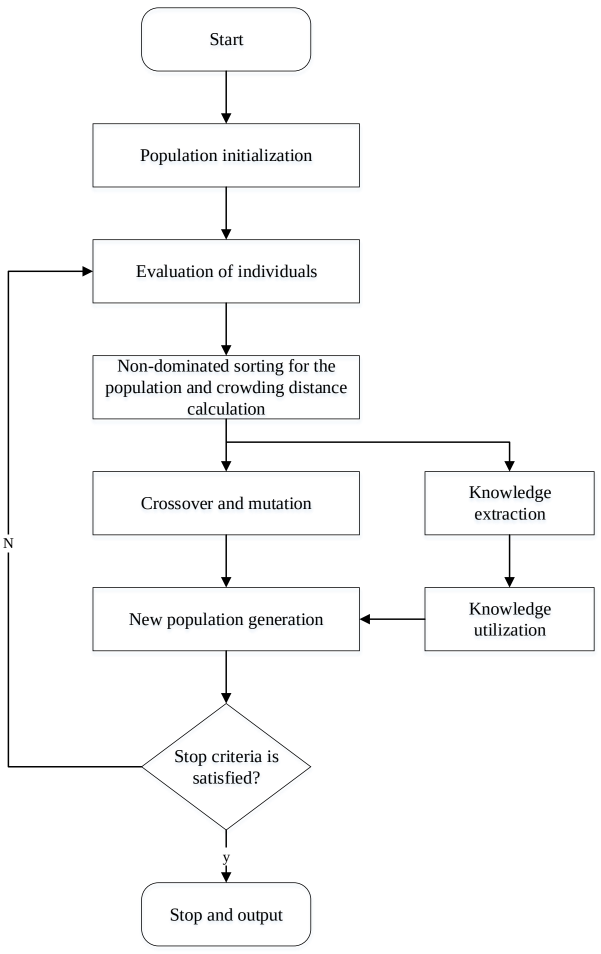

Start
Population initialization
Evaluation of individuals
Non-dominated sorting for the population and crowding distance calculation
Crossover and mutation
Knowledge extraction
N
New population generation
Knowledge utilization
Stop criteria is satisfied?
y
Stop and output

Figure 3. The overall workflow of KD-MOEA.

3.4.1. Knowledge Extraction

Knowledge is an abstract, logical thing that enables the process from quantitative to qualitative. Knowledge needs to be acquired through information inductive and deductive methods. In each generation of the population, the information contained in the individual exists in the form of vector data. The knowledge extraction process needs to organize these data and display them in a specific form. Specifically, in our problem, knowledge is divided into two categories, namely operation sequence knowledge and machine assignment knowledge, respectively shown in Figures 4 and 5.

a_{11}	a_{12}	……	a_{1N}
a_{21}	a_{22}	……	a_{2N}
……	……	……	
a_{N1}	a_{N2}	……	a_{NN}

Figure 4. The structure of operation sequence knowledge.

$(b_{11},b_{12},...,b_{1u_1})$	$(b_{21},b_{22},...,b_{2u_1})$	……	$(b_{N1},b_{N2},...,b_{2u_N})$

Figure 5. The structure of machine assignment knowledge.

We suppose there are N operations, i.e., N=|O|. In Figure 4, $a_{ij}(i, j = 1, 2, \cdots N)$ represents the times of ith operation appearing at the position j. In the representation of machine assignment knowledge, there are N vectors. For the ith vector $(b_{i1}, b_{i2}, \ldots, b_{iu_i})$, u_i is the number of machines that can process the operation in position i. $b_{ig}(g = 1, 2, \ldots, u_i)$ represents the times of assigning operation i to machine g.

It should be noted that operation sequence knowledge and machine assignment knowledge are stored in two structure tables, respectively. The data in the knowledge structure table comes from the non-dominated individuals in each generation of the population, that is, individuals with a non-dominated rank equal to 1. During each iteration, the operation sequence knowledge structure table and machine assignment knowledge structure table need to be updated once.

3.4.2. Knowledge Use

Operation sequence knowledge and machine assignment knowledge need to be used to guide subsequent optimization processes after extraction. In the KD-MOEA proposed in this paper, a population update strategy is used to use the obtained knowledge. In each population update process, when a new population is generated, individuals in the population are updated with a probability p_u. The new individuals are generated based on the operation sequence knowledge structure table and the machine assignment knowledge table. In addition, the values on the operation sequence chromosome and the machine assignment chromosome are generated by roulette. The population update operation is not performed every iteration, but a population update operation is performed every *t_gen* generations.

4. Experimental Analysis

4.1. Test Instances

To illustrate the addressed MOASPSP and test the proposed KD-MOEA, a set of test instances were constructed. The test set mainly includes machine data and task data. The machine data remains fixed, including a total of 99 machines with 21 types, as shown in Table 2. In the research, the personnel used in processing operations are regarded as a type of machine resource. It should be noted that the machine types in Table 2 are the machine types of the entire production workshop, and the machine types used in the test instances in this paper are a subset of the machine types of the entire workshop.

In the test instances, we considered four types of jobs, corresponding to shell type 1, 2, 3 and 4. Due to confidentiality reasons, the procedures of each type of shell are not shown in this paper. The test set contains six instances, as shown in Table 3. The elements in Table 3 indicate the job number for each type. For example, instance 1 contains four jobs, the numbers of different job types are the same as 1.

Table 2. Data of available machines in the workshop.

Machine Type	Machine Number	Unit Cost (10,000 RMB)
1	6	2300
2	5	2850
3	3	3100
4	4	3250
5	2	3750
6	3	3650
7	2	4000
8	3	3350
9	5	4200
10	4	4050
11	8	3900
12	8	2550
13	4	3450
14	4	2865
15	3	3674
16	4	3890
17	8	230
18	7	210
19	9	240
20	3	300
21	4	5300

Table 3. Data of test instance structure.

Instance Index	Shell Type 1	Shell Type 2	Shell Type 3	Shell Type 4
instance 1	1	1	1	1
instance 2	2	1	1	1
instance 3	2	2	1	1
instance 4	2	2	2	1
instance 5	2	2	2	2
instance 6	3	3	3	3

4.2. Parameter Settings

The baseline algorithm used in the KD-MOEA algorithm proposed in this paper is NSGA-II, and the algorithm comparison is compared with NSGA-II and MOEA/D. For the NSGA-II and MOEA/D algorithms, the population size is set to 100, the number of iterations is set to 200, and the probability of crossover and mutation is 0.9 and 0.1, respectively. Other parameters of the NSGA-II and MOEA/D algorithms are the same as those in [24] and [25]. For the KD-MOEA algorithm, the parameter *t_gen* is set as follows: *t_gen* = 10 in the first 100 iterations; *t_gen* = 5 in the last 100 iterations. The population update probability of KD-MOEA is set to $p_u = 0.1$. To avoid the randomness of the algorithms, each algorithm runs independently 30 times. All computers running the algorithm are configured as desktops with Core I7-7700, 3.6 GHz CPU, and the algorithm design is implemented using C#.

4.3. Experimental Results

4.3.1. Obtained Non-Dominated Solutions

This paper first analyzes the distribution of non-dominated solutions of the test instances. Figure 6 shows the distribution of the non-dominated solutions in the objective space obtained from the six test instances. It can be seen from Figure 6 that the approximating Pareto front of MOASPSP has strong discreteness and non-convexity. From the distribution of non-dominated solutions, there are also differences in the sparseness of the distribution across the entire non-dominated front. Specifically, when the cost of the production line increases, the non-dominated solution is more densely distributed;

when the cost of the production line is reduced, the non-dominated solution is more sparsely distributed. In other words, the greater the production line cost budget is, the more options are available. When production line costs are more severely constrained, fewer options are available.

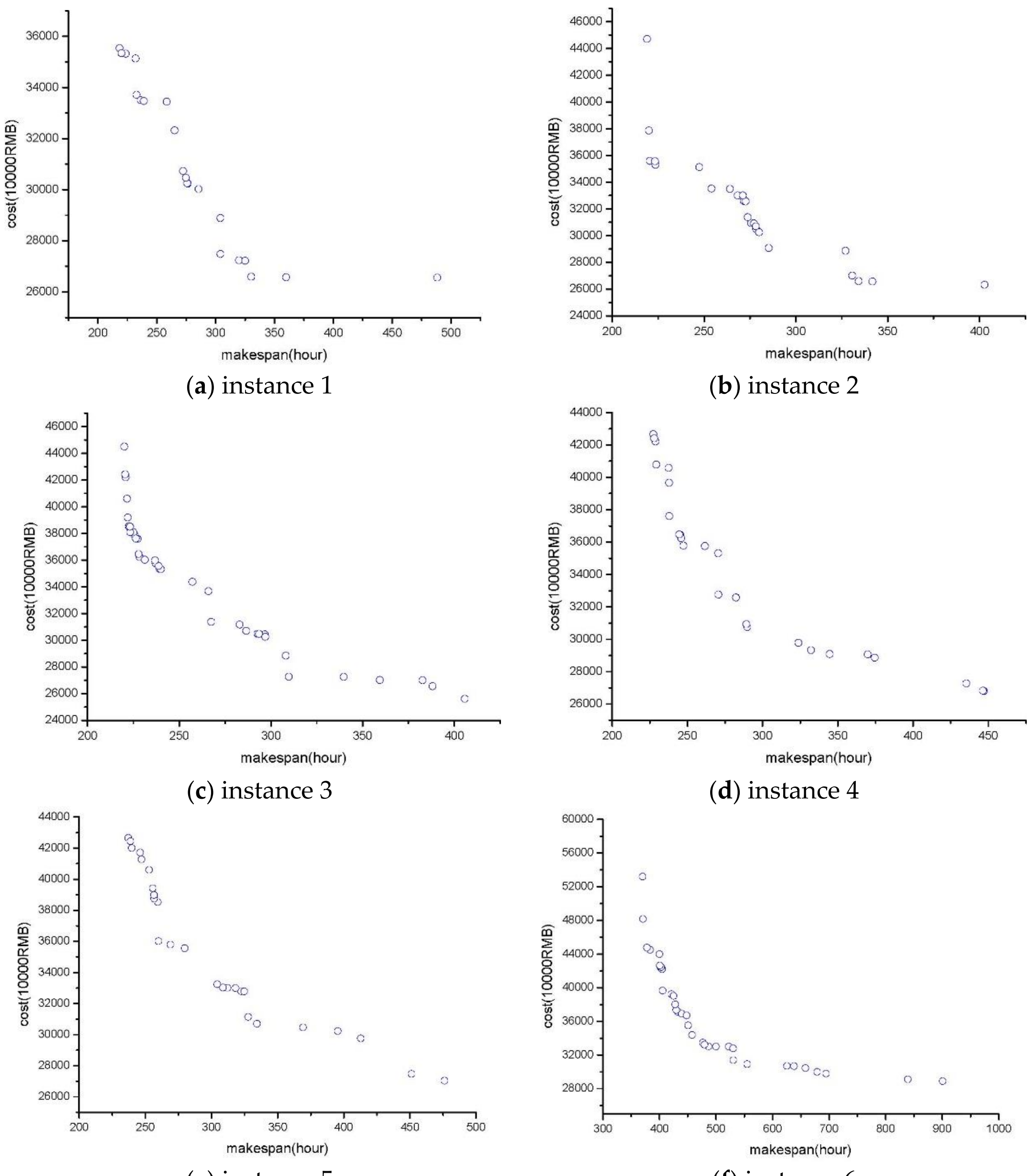


Figure 6. Distribution of non-dominated solutions in the objective space of different test instances.

To further study the structural differences of different non-dominated solutions, three solutions in instance 1 and instance 6 were selected for analysis. The selected solutions are shown in Table 4.

For each test instance, two non-dominated solutions at the boundary of the non-dominated frontier and one non-dominated solution in the middle are selected. From the perspective of the objective function, for the most economical production scheduling solutions, that is, the least production cost solutions such as solution 3 and solution 6, when the resource input is appropriately increased, the efficiency index of the production line in completing the processing tasks can be greatly improved. For example, for instance 1, when the cost of 920 units is added to the solution 3, the completion time can be increased by 184.1 units. If the indicators of completion time need to be further improved, a greater amount of resources need to be invested. For example, compared with solution2, solution 1 has an

85.7 unit completion time improvement and a production line cost of 8060 units. There is a similar phenomenon in instance 6.

Table 4. Selected non-dominated solutions from instance 1 and instance 6.

Solution	Instance	Makespan	Cost
solution 1	instance 1	218.5	35540
solution 2	instance 1	304.2	27480
solution 3	instance 1	488.3	26560
solution 4	instance 6	370.2	53190
solution 5	instance 6	530.1	32790
solution 6	instance 6	900.9	28900

It can be seen that the cost-effectiveness ratio of solutions for the aerospace product shell production line is quite different. In the actual production process, due to the constraints of resources and costs, and the requirements of completion time, often the feasible solution on the boundary will not be selected. It is necessary to select the feasible solution with the best cost-effective ratio according to the actual constraints and the availability of resources.

4.3.2. Performance Comparison

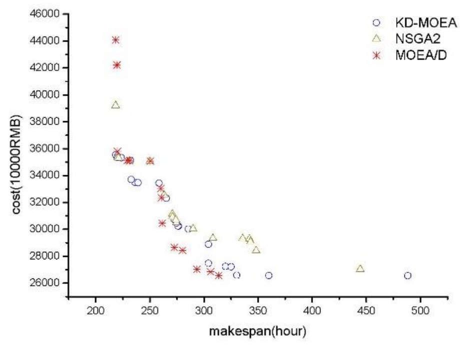
Performance comparison is an essential part of illustrating the effectiveness of an algorithm [27,28]. To analyze the performance of KD-MOEA proposed in this paper, the performance of the algorithm is compared in this subsection. The optimization results of KD-MOEA are compared with the optimization results of NSGA-II and MOEA/D algorithms. In the performance comparison of multi-objective optimization algorithms, the commonly used comparison index is the hypervolume (HV). Table 5 shows the HV values of the different algorithms on the six test instances. For the non-dominated solution obtained by the multi-objective optimization algorithm, the larger the HV value, the better. During the pairwise comparison of algorithms, a t-test was performed. When there is a significant difference in results, the results corresponding to algorithms with high performance are marked with (+), and the results corresponding to algorithms with poor performance are marked with (-). If there is no significant difference between the two, use the (~) mark.

Table 5. Comparison of HV results of different algorithms.

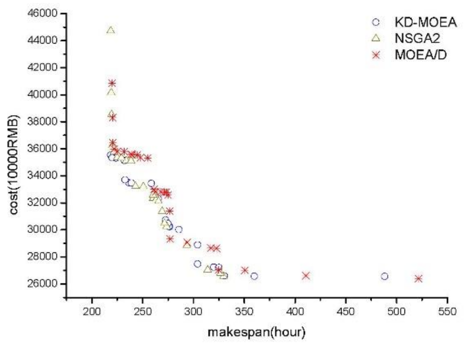
Instance	KD-MOEA	NSGA-II	MOEA/D
instance 1	0.7256 ± 0.0177 (+)	0.7140 ± 0.0149 (-)	0.7099 ± 0.0167 (-)
instance 1	0.7259 ± 0.0167 (+)	0.7195 ± 0.0178 (~)	0.7081 ± 0.0211 (-)
instance 1	0.7168 ± 0.0209 (+)	0.6950 ± 0.0215 (-)	0.6873 ± 0.0187 (-)
instance 6	0.6819 ± 0.0163 (+)	0.6722 ± 0.0205 (~)	0.6432 ± 0.0251 (-)
instance 6	0.6566 ± 0.0333 (+)	0.6332 ± 0.0183 (-)	0.6091 ± 0.0224 (-)
instance 6	0.4998 ± 0.0401 (+)	0.4487 ± 0.0495 (-)	0.4221 ± 0.0411 (-)

As can be seen from the above table, the KD-MOEA proposed in this paper outperforms MOEA/D on all six test instances. Compared with NSGA-II, the performance of KD-MOEA in four test cases (instance 1, instance 3, instance 5 and instance 6) is significantly better than NSGA-II. In the other two test instances, the statistical values are not significantly different. However, it can be seen that the HV of the non-dominated solution set obtained by KD-MOEA is greater than the corresponding value of the NSGA-II algorithm on all test instances. Therefore, this paper considers that KD-MOEA has a better effect than MOEA/D and NSGA-II when solving the MOASPSP. This also shows that when solving complex combinatorial multi-objective optimization problems, extracting and applying relevant knowledge based on the structure and characteristics of the problem under study will help improve the optimization performance of the algorithm and provide more scientific alternatives for decision-makers.

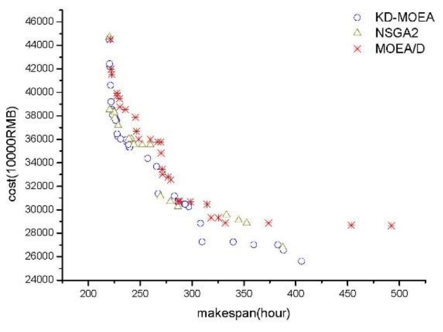
To more intuitively show the difference in performance of different algorithms, Figure 7 shows the distribution of non-dominated solutions in the objective space obtained by the three algorithms. As can be seen from Figure 7, for test instance 1, MOEA/D can obtain some better solutions, compared to KD-MOEA and NSGA-II. However, in terms of the spread of the non-dominated solution, that is, the distance between two boundary points, the performance of MOEA/D is worse than that of KD-MOEA and NSGA-II. It shows that in solving small-scale problems, the multi-objective evolutionary algorithm based on decomposition can approach the true non-dominated frontier in some areas, but at the same time sacrifices the diversity of the algorithm. As the problem size increases, KD-MOEA and NSGA-II are better than MOEA/D in terms of convergence and diversity. Compared with the proposed KD-MOEA and NSGA-II, the performance of the two algorithms in maintaining diversity is comparable, and the obtained non-dominated solutions have no significant difference in the spread of the objective space. As can be seen from Figure 7, most of the solutions obtained by NSGA-II are dominated by the solutions obtained by KD-MOEA, indicating that KD-MOEA has better performance in terms of algorithm convergence.



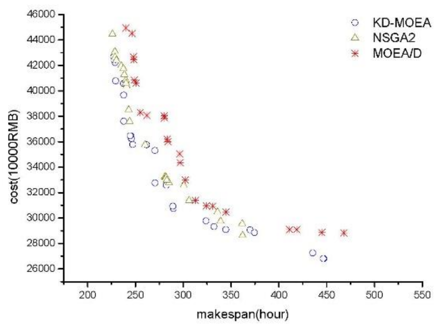
(**a**) instance 1



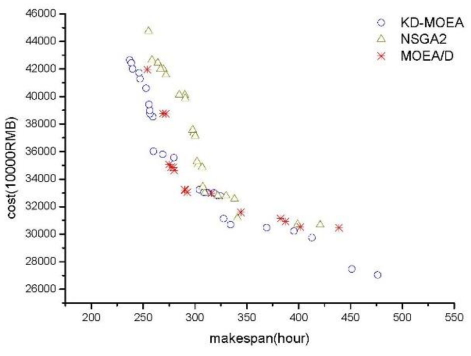
(**b**) instance 2



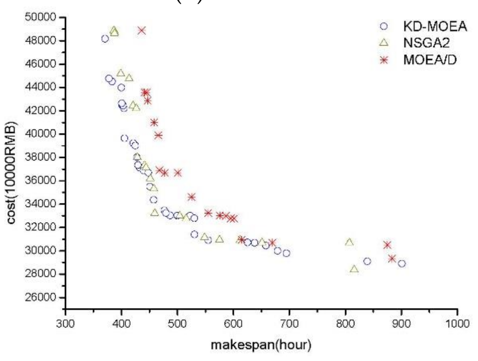
(**c**) instance 3



(**d**) instance 4



(**e**) instance 5



(**f**) instance 6

Figure 7. Distribution of non-dominated solutions with KD-MOEA, NSGA-II and MOEA/D (**a**) instance 1 (**b**) instance 2 (**c**) instance 3 (**d**) instance 4 (**e**) instance 5 (**f**) instance 6.

5. Conclusions

In this study, we investigated the aerospace shell product digital production line scheduling problem in an actual production process. Considering the efficiency and economic indicators of the production of aerospace product shells, a multi-objective optimization model for production scheduling is constructed. In the proposed model, one objective is the makespan of the production schedule and the other objective is the construction cost the production line. Aiming at the characteristics of the multi-objective optimization problem for aerospace shell production scheduling, a knowledge-driven multi-objective evolutionary algorithm was designed to solve the problem. Experimental results suggest that the proposed multi-objective approach has a better performance to solve the problem. The developed multi-objective optimization model and the proposed multi-objective optimization algorithm can effectively support the multi-objective operation of the aerospace shell production line. In the future, it is necessary to deeply explore the applicable rules of rescheduling, and build a dynamic production line scheduling model. To solve the production paralysis problem of different scales, analyze the impact of dynamic events on existing scheduling plans shall be investigated in future research.

Author Contributions: Conceptualization, Q.W. and J.X.; methodology, Q.W. and H.L.; software, X.W.; validation, J.X.; formal analysis, Q.W.; resources, J.X.; data curation, H.L.; writing—original draft preparation, Q.W., X.W., H.L. and J.X.; project ministration: J.X.; funding acquisition, J.X. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by National Natural Science Foundation of China under Grants 71501181 and 71871185.

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

References

1. Corominas, A.; Garcia-Villoria, A.; González, N.-A.; Pastor, R. A multistage graph-based procedure for solving a just-in-time flexible job-shop scheduling problem with machine and time-dependent processing costs. *J. Oper. Res. Soc.* **2018**, *70*, 620–633. [CrossRef]
2. Wu, X.; Liu, X.; Zhao, N. An improved differential evolution algorithm for solving a distributed assembly flexible job shop scheduling problem. *Memetic Comput.* **2018**, *11*, 335–355. [CrossRef]
3. Kavitha, S.; Venkumar, P.; Rajini, N.; Pitchipoo, P. An Efficient Social Spider Optimization for Flexible Job Shop Scheduling Problem. *J. Adv. Manuf. Syst.* **2018**, *17*, 181–196. [CrossRef]
4. Xie, N.; Chen, N. Flexible job shop scheduling problem with interval grey processing time. *Appl. Soft Comput.* **2018**, *70*, 513–524. [CrossRef]
5. Defersha, F.; Bayat-Movahed, S. Linear programming assisted (not embedded) genetic algorithm for flexible jobshop scheduling with lot streaming. *Comput. Ind. Eng.* **2018**, *117*, 319–335. [CrossRef]
6. Xu, Y.; Wang, L.; Wang, S.-Y.; Liu, M. An effective teaching–learning-based optimization algorithm for the flexible job-shop scheduling problem with fuzzy processing time. *Neurocomputing* **2015**, *148*, 260–268. [CrossRef]
7. Gao, K.Z.; Suganthan, P.; Pan, Q.-K.; Chua, T.; Chong, C.S.; Cai, T.-X. An improved artificial bee colony algorithm for flexible job-shop scheduling problem with fuzzy processing time. *Expert Syst. Appl.* **2016**, *65*, 52–67. [CrossRef]
8. Quan, K.P. An effective discrete harmony search algorithm for flexible job shop scheduling problem with fuzzy processing time. *Int. J. Prod. Res.* **2015**, *53*, 1–16.
9. Birgin, E.; Ferreira, J.; Ronconi, D.P. List scheduling and beam search methods for the flexible job shop scheduling problem with sequencing flexibility. *Eur. J. Oper. Res.* **2015**, *247*, 421–440. [CrossRef]
10. Geyik, F.; Dosdoğru, A.T. Process plan and part routing optimization in a dynamic flexible job shop scheduling environment: An optimization via simulation approach. *Neural Comput. Appl.* **2012**, *23*, 1631–1641. [CrossRef]
11. Duarte, J.L.R.; Fan, N.; Jin, T. Multi-process production scheduling with variable renewable integration and demand response. *Eur. J. Oper. Res.* **2020**, *281*, 186–200. [CrossRef]

12. Karner, M.; Glawar, R.; Sihn, W.; Matyas, K. An industry-oriented approach for machine condition-based production scheduling. *Procedia CIRP* **2019**, *81*, 938–943. [CrossRef]
13. Plinere, D.; Aleksejeva, L. Production scheduling in agent-based supply chain for manufacturing efficiency improvement. *Procedia Comput. Sci.* **2019**, *149*, 36–43. [CrossRef]
14. Paithankar, A.; Chatterjee, S. Open pit mine production schedule optimization using a hybrid of maximum-flow and genetic algorithms. *Appl. Soft Comput.* **2019**, *81*, 105507. [CrossRef]
15. Jélvez, E.; Morales, N.; Nancel-Penard, P.; Cornillier, F. A new hybrid heuristic algorithm for the Precedence Constrained Production Scheduling Problem: A mining application. *Omega* **2019**, 102046. [CrossRef]
16. Samavati, M.; Essam, D.; Nehring, M.; Sarker, R. Production planning and scheduling in mining scenarios under IPCC mining systems. *Comput. Oper. Res.* **2020**, *115*, 104714. [CrossRef]
17. Mai, N.L.; Topal, E.; Erten, O.; Sommerville, B. A new risk-based optimisation method for the iron ore production scheduling using stochastic integer programming. *Resour. Policy* **2019**, *62*, 571–579. [CrossRef]
18. Kaylani, H.; Atieh, A.M. Simulation Approach to Enhance Production Scheduling Procedures at a Pharmaceutical Company with Large Product Mix. *Procedia CIRP* **2016**, *41*, 411–416. [CrossRef]
19. Rawlings, B.C.; Avadiappan, V.; LaFortune, S.; Maravelias, C.T.; Wassick, J.M. Incorporating automation logic in online chemical production scheduling. *Comput. Chem. Eng.* **2019**, *128*, 201–215. [CrossRef]
20. Russell, A.; Taghipour, S. Multi-objective optimization of complex scheduling problems in low-volume low-variety production systems. *Int. J. Prod. Econ.* **2019**, *208*, 1–16. [CrossRef]
21. Lan, T.; Kao, D.; Chiang, M.; Sabharwal, A. An Axiomatic Theory of Fairness in Network Resource Allocation. In Proceedings of the 2010 IEEE INFOCOM, San Diego, CA, USA, 14–19 March 2010; pp. 1–9.
22. Zabini, F.; Bazzi, A.; Masini, B.M. Throughput versus fairness tradeoff analysis. In Proceedings of the 2013 IEEE International Conference on Communications (ICC), Budapest, Hungary, 9–13 June 2013; pp. 5131–5136.
23. Zhou, A.; Qu, B.-Y.; Li, H.; Zhao, S.-Z.; Suganthan, P.; Zhang, Q. Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm Evol. Comput.* **2011**, *1*, 32–49. [CrossRef]
24. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T.A.M.T. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [CrossRef]
25. Zhang, Q.; Li, H. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Trans. Evol. Comput.* **2007**, *11*, 712–731. [CrossRef]
26. Wang, Q.; Luo, H.; Xiong, J.; Song, Y.-J.; Zhang, Z. Evolutionary Algorithm for Aerospace Shell Product Digital Production Line Scheduling Problem. *Symmetry* **2019**, *11*, 849. [CrossRef]
27. Zhang, Z.; Hong, W.-C.; Li, J. Electric Load Forecasting by Hybrid Self-Recurrent Support Vector Regression Model With Variational Mode Decomposition and Improved Cuckoo Search Algorithm. *IEEE Access* **2020**, *8*, 14642–14658. [CrossRef]
28. Hong, W.-C.; Li, M.-W.; Geng, J.; Zhang, Y. Novel chaotic bat algorithm for forecasting complex motion of floating platforms. *Appl. Math. Model.* **2019**, *72*, 425–443. [CrossRef]

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).