IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received December 28, 2019, accepted January 16, 2020, date of publication January 21, 2020, date of current version January 30, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2968378

Efficiency-Oriented Production Scheduling Scheme: An Ant Colony System Method

YI WAN[1,2,3,4], (Member, IEEE), TIAN-YU ZUO[4], LIN CHEN[2], WEN-CHENG TANG[3,4], AND JIN CHEN[4,5]

[1]School of Environmental Science, Nanjing Xiaozhuang University, Nanjing 211171, China
[2]Jiangsu Shentong Valve Company, Ltd., Nantong 226232, China
[3]School of Mechanical Engineering, Southeast University, Nanjing 211189, China
[4]School of Mechanical and Electrical Engineering, Sanjiang University, Nanjing 210012, China
[5]School of Astronautics, Nanjing University of Aeronautics and Astronautics, Nanjing 211189, China

Corresponding author: Tian-Yu ZUO (tian_yu_zuo@163.com)

This work was supported in part by the Natural Science Foundation of the Jiangsu Higher Education Institutions of China under Grant 19KJB460006, and in part by the China Postdoctoral Science Foundation under Grant 2019M651642.

ABSTRACT During the real production system, the scheduling scheme change is mostly changed by dynamic events or new tasks. Due to the different urgency degrees of dynamic events, the corresponding scheduling methods should be adopted to ensure the production efficiency of enterprises. In this paper, an event-driven dynamic workshop scheduling model is established based on Ant Colony System (ACS), and two scheduling methods are designed to deal with dynamic events, namely parallel scheduling and parallel priority scheduling, respectively. The goal of parallel scheduling is to minimize the total makespan, while that of parallel priority scheduling is to minimize the delivery time of dynamic events. Additionally, a selective scheduling strategy is designed to determine the optimal scheduling method according to the urgency degree of dynamic events. Finally, the feasibility of the selective scheduling strategy in solving the dual-objective dynamic job shop scheduling problem (DJSP) is verified by an example experiment on DJSP as well as a large scale problem test set.

INDEX TERMS Job shop scheduling, ant colony system, dynamic scheduling, event-driven.

I. INTRODUCTION

Job Shop Problem (JSP) [1] is a hot topic in the study of production scheduling. JSP is the optimization of grouping problem, whose goal is a reasonably arrangement of the processing sequence and starting time during each processing machine, and ensuring a certain target (such as the makespan) to meet the requirements, which has been proved to be a typical NP-hard [2]–[5]. At present, the heuristic optimization algorithms such as Ant Colony Optimization (ACO) [6], Genetic Algorithm (GA) [7], [8], Simulated Annealing Algorithm (SAA) [9], Particle Swarm Optimization (PSO) [10], [11], Tabu Search (TS) [12], [13], Gravitational Search Algorithm (GSA) [14] and Differential Evolution (DE) [15], that are widely used to solve JSP.

The idea of ACO is derived from the process of ants searching for food in nature. They communicate with each other indirectly to achieve group cooperation through a substance called pheromone, so as to find the shortest path from the colony to the food source. Through the abstract modeling of this swarm intelligence behavior, Dorigo [16] proposed the ACO in 1992. ACO has good robustness, strong search ability and positive feedback mechanism, so it is widely used to solve optimization problems, especially combinatorial optimization problems, such as traveling salesman problem(TSP) [17], knapsack problem, vehicle path problem (VRP) [18]–[20]. In addition, ACO has been successfully applied to JSP. Colorni [21] applied ACS to JSP for the first time, but the algorithm needs to be optimized. Heinonen and Pettersson [22] applied a hybrid ant colony algorithm to solve the benchmark scheduling problem MT10. Blum and Sampels [23] proposed a neighborhood structure for this problem by extending the neighborhood structure derived from Eugeniusz and Czeslaw [24], which improves the algorithm solving ability and is the first competitive ant colony optimization approach in JSP instances. In 1997, Dorigo and Gambardella [25] proposed the ACO with a new mechanism, which was a milestone work in the

The associate editor coordinating the review of this manuscript and approving it for publication was Hao Luo.

development history of ACO with better performance than ACO.

Graves [26] classified and sorted out JSP. The research on the common JSP that simplifies the situation of the actual production is no longer the key research object. Insteadly, the research is the more complex and closer to the actual production of JSP, such as dynamic JSP, flexible JSP. On the basis of solving the static JSP, the dynamic JSP also needs to deal with the dynamic interference in the real processing environment, and constantly re-scheduling according to the actual situation.

For the flexible JSP, Yan [27] and Huang *et al.* [29] proposed an improved ant colony algorithm to solve the flexible JSP and successfully solved the ability-constrained flexible job-shop problem using ant colony genetic algorithm and used ant colony algorithm to solve the flexible job shop problem (FJSP-MPP) on multi-processing route.

Mohan *et al.* [30] broadly summarized the development of DJSP. Li *et al.* [31] advanced an adaptive ACO solution method of DJSP, and improved the global search ability of the algorithm. Chen *et al.* [32] introduced the multi-agent system based on ACO to solve DJSP, and implemented the scheduling of dynamic events in the system.

At present, the research on DJSP mainly realizes the ability of solving dynamic events, and most of them are single-objective optimization. But the actual production process is complex. Therefore, it is of great practical significance and value to study how to adjust and select a reasonable scheduling method to ensure the overall efficiency of the enterprise in real time according to the workshop situation. Zhang and Dang [33] proposed a data-driven dynamic scheduling strategy and designed a scheduling strategy to adjust the scheduling scheme according to the real-time situation of the workshop, contributing to the research on selective scheduling. Suresh [34] classified dynamic events into four types:

1. Events related to work pieces: random arrival of work pieces, uncertain processing times, and order changes, etc.
2. Machine-related events: machine damage, machine blocking/deadlock, and production capacity conflict, etc.
3. Process related events: process delay, output instability, etc.
4. Other incidents: worker absenteeism, lack of raw materials, etc.

The DJSP is studied based on events related to workpieces in this paper, which targets at makespan and the minimum of timeout punishment based on the addition of new orders as the only dynamic events. A selective scheduling strategy based on ACS [35] is developed to ensure the overall efficiency of enterprises by choosing the most optimal scheduling method according to the urgency degree and actual scheduling situation, and make the enterprise have the ability to handle dynamic tasks and ensure the comprehensive benefits.

The rest of the paper is as follows: Section II describes the JSP in mathematical language; Section III defines the mathematical model of the selective scheduling scheme; Section IV is the algorithm design according to the mathematical model established by Section III; Section V performs the simulation experiment. Finally, the conclusion will be shown in Section VI.

II. DESCRIPTION OF PROBLEM

A. CLASSIFICATION OF JSP

1) STATIC JSP

The classic JSP is a kind of practical production process of simplification, mainly to solve the static environment of the job shop scheduling problem. The actual production process should be dynamic and complex, such as new orders to join, the original order cancellation, machine damage, raw materials delayed arrival and so on.

Suppose that there are m machines and n workpieces to be machined in the workshop. The goal of scheduling is to find an optimal processing sequence that satisfies the objective function.

Processing machine number of the process j of the workpiece i is denoted by M_{ij}, and the workpiece set can be represented by the workpiece matrix M of $n*m$:

$$M = \begin{bmatrix} M_{11} & M_{12} & \cdots & M_{1m} \\ M_{21} & M_{22} & \cdots & M_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ M_{n1} & M_{n2} & \cdots & M_{nm} \end{bmatrix} \tag{1}$$

The processing time required for the process j of the workpiece I is represented by T_{ij}, and the time set can be represented by a time matrix T of $n*m$:

$$T = \begin{bmatrix} T_{11} & T_{12} & \cdots & T_{1m} \\ T_{21} & T_{22} & \cdots & T_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ T_{n1} & T_{n2} & \cdots & T_{nm} \end{bmatrix} \tag{2}$$

Refer to the definition of the mathematical variables [36]: R_{ijegk} and X_{ijk} are the binary coefficients. P_{ij}— the j process of the workpiece i. P_{eg}— the g process of the workpiece e. T_{ijk}— time spent by P_{ij} on machine k; S_{ijk}— starting time of P_{ij} processing on machine k; E_{ijk}— ending time of P_{ij} processing on machine k. When the value of R_{ijegk} is 1, the operation j of workpiece i and the operation g of workpiece e need to be carried out on machine k. The coefficient of X_{ijk} indicates whether the operation j of workpiece i needs to be carried out on machine k. When the value is 1, it is required. Otherwise, it is not required.

$$R_{ijegk} = \begin{cases} 1, & P_{ij} \text{ and } P_{eg} \text{ use the same machine k} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

$$X_{ijk} = \begin{cases} 1, & \text{if } P_{ij} \text{ use the machine k} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

The constraints on the problem are as follows:

$$\begin{cases} E_{ijk} - E_{i(j-1)k} \geq T_{ijk} 1) \\ E_{egk} - E_{ijk} \geq T_{egk} 2) \end{cases} \tag{5}$$

Among above, $X_{ijk} = X_{egk} = 1$, $R_{ijegk} = 1$, $X_{ijk} = X_{i(j-1)k} = 1$.

In the formula (5), 1) indicates that the processing of the workpiece must follow the sequence of working procedure strictly. That is to say, P_{ij} must be finished before $P_{i(j-1)}$ can be started. 2) means that machine k can only process one piece at a time.

The rest of the assumptions for the problem are as follows:

1. Processing tasks must not be interrupted.
2. The priorities between the workpieces are the same.
3. At time 0, all workpieces can be machined.
4. The processing time of each process has been determined and remains unchanged during the process.

2) DYNAMIC JSP

With the rest of the conditions unchanged, the workpiece matrix M(same as formula 1) and the time matrix T(same as formula 2) in the DJSP will change with the addition of dynamic events. There are n workpieces to be processed in the workshop at time 0. At time t new tasks ($P_{(n+1)1}, P_{(n+1)2}, \ldots, P_{(n+1)m}$) are added, the workpiece matrix M and the time matrix T change accordingly into M' and T':

$$\mathrm{M} = \begin{bmatrix} \mathrm{M}_{11} & \mathrm{M}_{12} & \cdots & \mathrm{M}_{1\mathrm{m}} \\ \mathrm{M}_{21} & \mathrm{M}_{22} & \cdots & \mathrm{M}_{2\mathrm{m}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{M}_{\mathrm{n}1} & \mathrm{M}_{\mathrm{n}2} & \cdots & \mathrm{M}_{\mathrm{nm}} \end{bmatrix} \Rightarrow \begin{bmatrix} \mathrm{M}_{11} & \mathrm{M}_{12} & \cdots & \mathrm{M}_{1\mathrm{m}} \\ \mathrm{M}_{21} & \mathrm{M}_{22} & \cdots & \mathrm{M}_{2\mathrm{m}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{M}_{\mathrm{n}1} & \mathrm{M}_{\mathrm{n}2} & \cdots & \mathrm{M}_{\mathrm{nm}} \\ \mathrm{M}_{(\mathrm{n}+1)1} & \mathrm{M}_{(\mathrm{n}+1)2} & \cdots & \mathrm{M}_{(\mathrm{n}+1)\mathrm{m}} \end{bmatrix} \tag{6}$$

$$\mathrm{T} = \begin{bmatrix} \mathrm{T}_{11} & \mathrm{T}_{12} & \cdots & \mathrm{T}_{1\mathrm{m}} \\ \mathrm{T}_{21} & \mathrm{T}_{22} & \cdots & \mathrm{T}_{2\mathrm{m}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{T}_{\mathrm{n}1} & \mathrm{T}_{\mathrm{n}2} & \cdots & \mathrm{T}_{\mathrm{nm}} \end{bmatrix} \Rightarrow \begin{bmatrix} \mathrm{T}_{11} & \mathrm{T}_{12} & \cdots & \mathrm{T}_{1\mathrm{m}} \\ \mathrm{T}_{21} & \mathrm{T}_{22} & \cdots & \mathrm{T}_{2\mathrm{m}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{T}_{\mathrm{n}1} & \mathrm{T}_{\mathrm{n}2} & \cdots & \mathrm{T}_{\mathrm{nm}} \\ \mathrm{T}_{(\mathrm{n}+1)1} & \mathrm{T}_{(\mathrm{n}+1)2} & \cdots & \mathrm{T}_{(\mathrm{n}+1)\mathrm{m}} \end{bmatrix} \tag{7}$$

The insertion of dynamic events will interfere with the original scheduling and affect the stability of the scheduling system. According to Song and Zhao [37], the rolling time can domain optimization strategy to solve the dynamic scheduling, and a rescheduling algorithm proposed by Abumizar and Svestka [38]. When dynamic interference occurs, the scheduling is not completely restarted, but only those that are disturbed. The dynamic event arrival time is set as t, and the unexecuted process at time t with the newly added process is combined to form a new workpiece M' and time set T'. We reschedule the new set to get the scheduling result containing the dynamic event.

B. COMPARATIVE ANALYSIS OF JSP

Static scheduling is an activity in which the scheduling task and its production factors (such as task number, processing time, etc.) remain unchanged. But the plan needs to be readjusted when the plan task elements change. Therefore static JSPS do not adequately reflect the actual production situation.

Dynamic scheduling has an ability to deal with random events in scheduling, which is different from static scheduling. When the scheduling task is interfered by random events, it can self-adjust and has high anti-jamming ability. Compared with static scheduling, it is more in line with the actual situation of the production site and adapted to the changeable and complex actual production environment. Therefore, it has a higher practical significance.

III. EMERGENCY DRIVEN ALTERNATIVE DJSP SCHEDULING STRATEGY

According to Section II, the general way to deal with the DJSP can be simply described as: combining the process that the original task has not been executed at time t with the new task added at time t, and scheduling it in parallel. Parallel scheduling can guarantee the minimum makespan, but this single processing mode cannot meet the actual production requirements for new tasks with different urgency. This paper presents a parallel priority scheduling method for dynamic events, which is called parallel priority for short. Selective DJSP scheduling strategy will be based on the urgency of dynamic events to choose the most optimal scheduling way to meet the objective function.

A. CLASSIFICATION OF URGENCY DEGREE

In the production of actual workshop, the new task (order) put forward by customer often has different degree of urgency. For new tasks of different urgency, enterprises should take different measures to ensure production efficiency under the premise of reasonable scheduling. Because the original task has signed a contract agreement, the due date for the original task (OT) cannot be changed. Define the due date for the new task (NT) as follows:

$$\mathrm{NT} = \mathrm{t} + \sum_{\mathrm{j}=1}^{\mathrm{m}} \mathrm{T}_{(\mathrm{n}+1)\mathrm{j}} + \lambda \times (10\% \times \sum_{\mathrm{j}=1}^{\mathrm{m}} \mathrm{T}_{(\mathrm{n}+1)\mathrm{j}}) \tag{8}$$

The λ is the urgency degree coefficient. According to formula (8), we can know that the new task delivery time T_2 is related to the value λ. For new tasks with different degrees of urgency, the delivery time T_2 should be proportional to the value λ of the urgency factor. This article divides the urgency of a new order into three categories and defines the urgency coefficient λ as:

$$\lambda = \begin{cases} 1.0, & \text{Immediate emergency} \\ 1.5, & \text{General emergency} \\ 2.0, & \text{Non} - \text{emergency} \end{cases} \tag{9}$$

B. SELECT OBJECTIVE FUNCTION FOR SCHEDULING

The goal of the selective scheduling strategy is to choose the most optimal suitable method according to the urgency of the current new task and the overall benefit of the enterprise. Among them, the indicators to judge the overall benefit are to minimize the makespan (including dynamic events) and complete the orders within the delivery period. However, the dual-objective function tends not to be satisfied at the same time. When the order cannot be completed within the delivery period, the goal of scheduling should ensure the minimum penalty cost. The original task penalty cost C1 and the new task penalty cost C2 are defined as follows:

$$\begin{cases} C_1 = OT - \max\left(E_{ijk}\right), & i \in (1, n+1)\ j, k \in (1, m) \\ C_2 = NT - \max\left(E_{(n+1)jk}\right), & j, k \in (1, m) \end{cases} \tag{10}$$

C. EMERGENCY TASK HANDLING STRATEGY

In the process of urgent task, there are parallel scheduling and parallel priority scheduling. The difference between parallel scheduling and parallel priority lies in the scheduling attitude to the new task. The parallel scheduling strategy is mainly applied to the tasks without priority and ensures the shortest completion time. In the scheduling arrangement, the new task is carried as the first object that means the new task has a scheduling priority on the machine after the arrival of the new task. The goal of parallel priority scheduling is to minimize the makespan of new tasks, but it will affect the scheduling results of total tasks.

In summary, the two scheduling methods have their own advantages and disadvantages, a single choice of a scheduling method cannot meet the production requirements. Therefore, it is the safest way to choose the scheduling strategy according to the actual situation of the workshop and the urgency of the new task. In this paper, new tasks with different degrees of urgency are processed as shown in Fig.1.

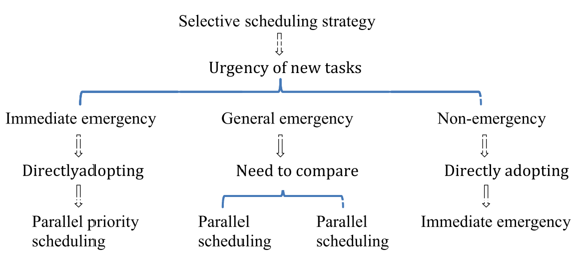

FIGURE 1. Scheduling for handling emergencies.

The processing method is described as follows:

1. For very urgent new tasks, the enterprise should give priority to new tasks and ensure the earliest completion of new tasks. So, choose parallelism first.

2. For general emergency new tasks, the enterprise chooses the most optimal scheduling method based on the comprehensive consideration of the actual situation.

3. For new tasks that are non-emergency, the enterprise does not need to prioritize new tasks, so parallel scheduling is adopted to ensure the makespan is the minimum.

Algorithm 1 Alternative Scheduling Strategy for General Emergency

```
Begin
  Calculate parallel scheduling;
   if Original task delivery time ≤ OT and New task
delivery time ≤ NT
     Employ parallel scheduling;
  else
C1 ← Penalty calculation;
C2 ← Penalty calculation;
C ← C1 + C2;
     Calculate parallel priority scheduling;
if Original task delivery time ≤ OT
       Employ parallel priority scheduling;
else
C1 ← Penalty calculation;
if C ≤ C1
         Employ parallel scheduling;
else
         Employ parallel priority scheduling;
end if
end if
  end if
End
```

The judgment process of selective scheduling for general urgent tasks is complex, and the pseudo-code of the judgment program can be found in Alg.1.

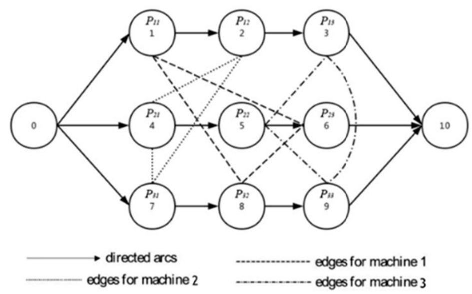

FIGURE 2. 3×3 JSP disjunctive graph.

IV. ALGORITHM DESIGN

A. APPLICATION OF ACS TO JSP

1) STRUCTURE OF THE SOLUTION [39]

ACS can be used to solve the TSP, and the disjunctive graph model is established to analyze this problem.

$$G = (V, A, E) \tag{11}$$

where

- V: is a set of nodes.
- A: is a set of conjunctive directed arcs.
- E: is a set of disjunctive, undirected edges.

The disjunctive graph is shown in Fig.2:

Let P = $\{P_{11}, P_{12}, \cdots, P_{nm}\}$, $(i, j) \in I, I \subseteq [1, n] \times [1, m]$ be the set of nodes. V, A and E are defined as follows:

- $V = P \cup \{P_0\} \cup \{P_{N+1}\}$, $\{P_0\}$ and $\{P_{N+1}\}$ are the special dummy nodes which identify the start and the completion of the overall Job–Shop).
- $A = \{(P_{ij}, P_{i(j+1)})$:$P_{ij} \rightarrow P_{i(j+1)}$ is in the chain for job $i\} \cup \{(P_0, P_{1j})$: P_{1j} is the first operation in the chain for job $i\} \cup \{(P_{im}, P_{N+1})$: P_{im} is the last operation in the chain for job $i\}$.
- $E = \{(P_{ij}, P_{hj})$, for all i.

The goal of the JSP is to find a process order that minimizes the makespan. Similar to the TSP, the ant starts from P_0, travels through all nodes and finally reaches P_{N+1} (each node is allowed to do so only once). The ant's walking path is a solution of the problem, and the ant's walking routing is the process of constructing the solution.

2) PROBABILITY TRANSFER

To facilitate the description of the problem, nodes i, j ($i \neq j$) are used to represent any two operable nodes in the disjunction map. The transfer of ants from node i to node j will be based on a pseudo random proportional rule:

$$j = \begin{cases} \arg\max_{j \in J_k} \left\{\tau(i,j)^{\alpha}, \eta(i,j)^{\beta}\right\}, & \text{if } q \leq q_0 \\ S, & \text{otherwise} \end{cases} \quad (12)$$

where

- J_k: a collection of nodes that are allowed to access, $J_k \subseteq A$.
- $\tau(i, j)$: the pheromone values of node i and node j.
- $\eta(i, j)$: the heuristic information of node i and node j, whose value is the reciprocal of the processing time required by node j.

The q_0 is a parameter within the interval of [0,1]. When the pseudo random number $q \leq q_0$, the ant will directly select the next node with the largest product of the heuristic information and the pheromone value, which is called 'development'. On the contrary, when the pseudo random number $q > q_0$, the next node will be selected according to the transfer probability formula (12) and the combination of Roulette selection strategy to choose the next node, called 'biased exploration'.

$$P_k(i,j) = \begin{cases} \dfrac{\left[\tau(i,j)^{\alpha}\right]\left[\eta(i,j)^{\beta}\right]}{\sum\limits_{u \in J_k}\left[\tau(i,u)^{\alpha}\right]\left[\eta(i,u)^{\beta}\right]}, & \text{if } j \in J_k \\ 0, & \text{otherwise} \end{cases} \quad (13)$$

where

- α: the importance of pheromone values in algorithms.
- β: the importance of heuristic information in algorithms.

By changing the size of q_0, the balance between 'development' and 'exploration' can be effectively adjusted, and the algorithm is determined whether to focus on developing the area near the optimal path or to explore other areas. Therefore, the quality of each parameter selection will directly affect the solving ability of the algorithm. Ye and Zheng [40] studied the setting of each parameter in the ACS and gave the recommended value of each parameter through experiments. At present, the trial-and-error method is a general method for parameter setting. In this paper, a satisfactory parameter setting is obtained by combining the recommended parameter values and trial and error experiments.

3) PHEROMONE UPDATE

a: GLOBAL UPDATE STRATEGY

Using the idea of ACS [41]–[43], the pheromone update is performed on the path constructed by the optimal solution to date at the end of each algorithm iteration, including the enhancement and volatilization of the pheromone. The rules are as follows:

$$\begin{cases} \tau(i,j) = (1-\rho) \cdot \tau(i,j) + \rho \cdot \Delta\tau(i,j), & \forall (i,j) \in \text{BestPath} \\ \Delta\tau(i,j) = \dfrac{1}{T_a} \end{cases} \quad (14)$$

where

- *BestPath*: path of global optimal solution.
- ρ: pheromone volatile factor.
- T_a: the total time of Ant a passing the path.

The strategy of pheromone update only for the optimal solution path helps to better guide the ant search bias and make the algorithm converge faster.

b: LOCAL UPDATE STRATEGY

In the process of ant construction, it is accompanied by the volatilization of pheromones. That is, each time an ant selects a node, from node i to node j, the pheromone is volatilized. The rules are as follows:

$$\tau(i,j) = (1-\xi) \cdot \tau(i,j) + \xi \cdot \tau_0 \quad (15)$$

where

- ξ: pheromone local volatilization factor.
- τ_0: pheromone initial value.

The pheromone local update rule can more realistically simulate the volatilization of the pheromone in the actual situation, and can reduce the probability that the edge of node i to node j is selected by other ants. By weakening the probability of selecting the same path among the ants, the ants tend to select the nodes that have not been selected, which can effectively improve the exploration ability of the algorithm and reduce the possibility that the algorithm falls into the local optimal solution.

B. SELECTIVE DJSP SCHEDULING ALGORITHM FLOW BASED ON ACS

The pseudo code of the selective DJSP scheduling algorithm can be found in Alg.2. The main flow of the selective DJSP scheduling algorithm is shown in Fig.3.

V. EXPERIMENT SIMULATION

The algorithm of this paper is tested to verify the feasibility of the algorithm. The algorithm uses M encoding and runs on a PC with Intel 3.6GHz i7-7700CPU and 4GB RAM under Windows 10 using Matlab2018a.

Algorithm 2 Selective DJSP Scheduling Strategy

```
Begin
Parameter initialization;
Set each matrix;
Initialize J_k;        //J_k is tabu matrix
a ← 1;                 //a is ant population
i ← 0;                 //Virtual node 0
iter ← 1;
NodeNum ← (n × m);     //NodeNum is total number of nodes
while iter ≤ iter_max do
 for a ← 1 to AntNum
 for i ← 0 to (NodeNum-1)
 j ← Use pseudorandom proportional rule;      //j ∈ J_k
 Pheromon_ij local update;                    //Update the pheromone on nodes i to j
 Process J_k, StartTime_{n×m} and EndTime_{n×m}; //Set the matrix from node i to node j and J_k
 end for
 end for
 BestTime[iter] ← Contemporary optimal solution makespan;
 BestRoute[iter] ← Contemporary optimal solution route;
 Pheromone global update;
 iter ← iter+1;
 According to BestTime and BestRoute get globally optimal solution; //Get the optimal solution of JSP
end while
Load dynamic event information;
According to t process matrix;                 //t is Dynamic event arrival time
According to λ perform selective scheduling strategy;
Execute optimal scheduling method;
Get the optimal solution under the optimal scheduling method;
End
```

TABLE 1. Parameters setting of ACO.

Parameter	Value	Description
iter_max	*150*	*Maximum number of iterations*
AntNum	*75*	*Ant number*
α	*2.0*	*Pheromone weight*
β	*1.0*	*Heuristic information weight*
ρ	*0.1*	*Pheromone volatile factor*
ξ	*0.15*	*Pheromone local volatilization factor*
q_0	*0.1*	*Pseudorandom number*

A. STATIC JSP SIMULATION EXPERIMENT

In this paper, the problem is taken as the experiment object. The parameters in the algorithm are shown in Table 1:

The iterative curve of the algorithm for solving the FT06 benchmark problem is shown in Fig.4 Compared with the literature [44], we can see that the proposed algorithm can better complete the scheduling of static JSP.

Observing the running time curve of different problem scales is not difficult to find that as the scale of the problem increases, the growth trend of the running time is linear, and the larger the problem size, the running time is a little long. Because the ASC has the ability to perform parallel computing the ACS has an advantage in dealing with large-scale practical problems. In this paper, the selective scheduling strategy designed by ACS has the ability to achieve results in large-scale practical problems.

TABLE 2. Workpiece matrix of dynamic event.

Process ID	P_{71}	P_{72}	P_{73}	P_{74}	P_{75}	P_{76}
Machine ID	2	3	1	4	5	6

TABLE 3. Processing time of dynamic events.

Test set	λ	T_{71}	T_{72}	T_{73}	T_{74}	T_{75}	T_{76}	Total Time
1	1	4	6	4	11	9	10	44
2	2	9	2	12	5	9	7	44
3	1.5	10	2	6	8	4	10	40
4	1.5	8	4	5	2	12	10	41
5	1.5	3	7	5	13	8	10	46
6	1.5	9	1	6	4	12	10	42

TABLE 4. Delivery date of dynamic event.

Test set	λ	OT	NT
1	1	75	68.4
2	2	75	72.8
3	1.5	75	66
4	1.5	75	67.15
5	1.5	75	72.9
6	1.5	75	68.3

B. SELECTIVE DJSP SCHEDULING STRATEGY SIMULATION EXPERIMENT

1) EXPERIMENT TEST SET SETTING

For the dynamic event arrival time $t=20$, the original task delivery time OT $= 75$ is taken as an example. In this paper, six dynamic tasks with the same workpiece set different urgency degree and processing time are set as test sets to verify the results of the algorithm when facing different dynamic tasks. The six test sets are shown in Table 2, 3, and 4:

2) EXPERIMENT RESULT

In Test Set1 to Test Set 6, 'None' means the scheduling method is not executed, 'OTC' means the original task completion time, 'NTC' means new task completion time.

- Test set 1 experiment results

Set 1 ($\lambda = 1$) is immediate emergency. According to the algorithm flow, the parallel priority scheduling should be adopted. The experiment results are as follows:

- Test set 2 experiment results

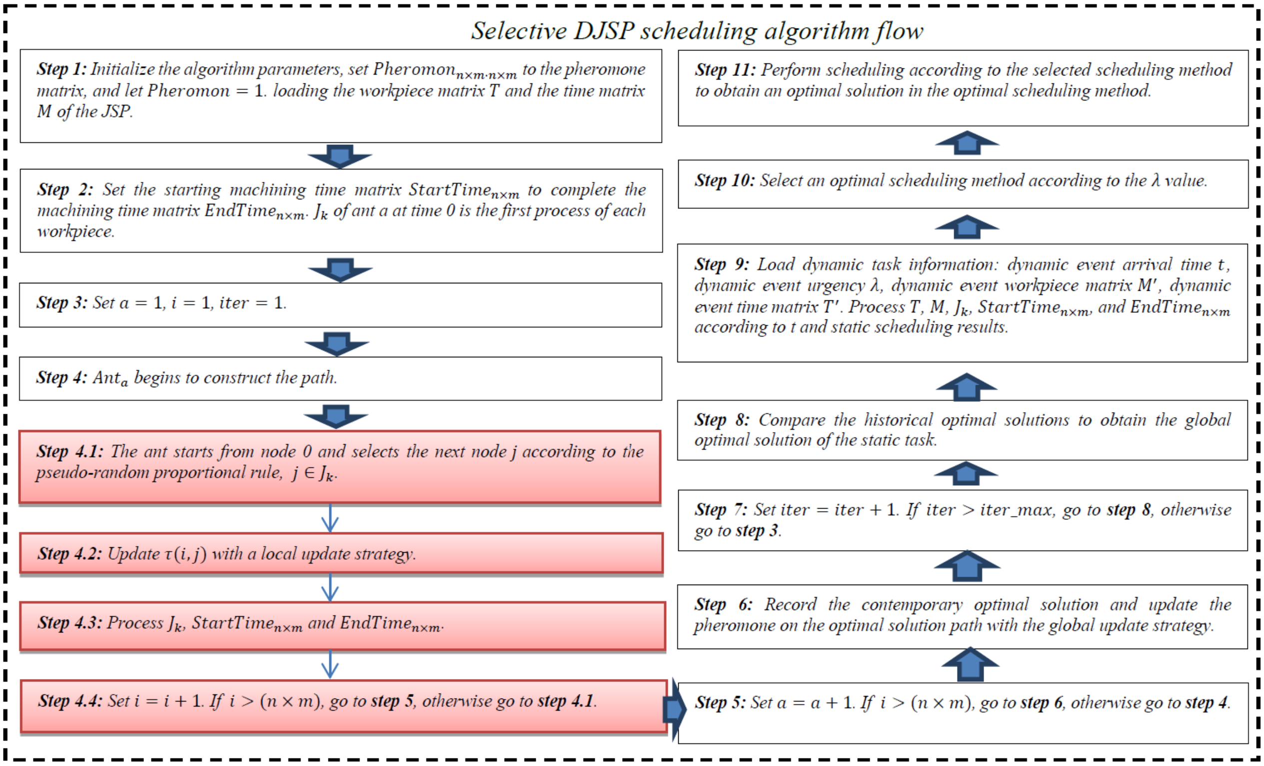

FIGURE 3. Selective DJSP scheduling algorithm flow.

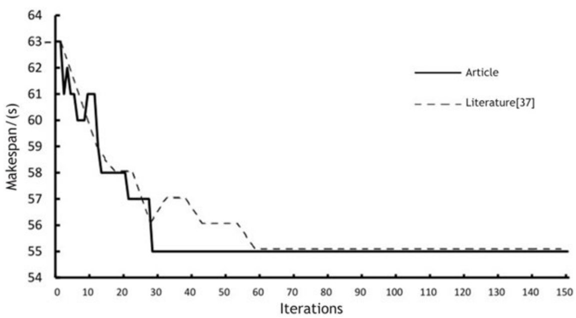

FIGURE 4. FT06 iteration curve.

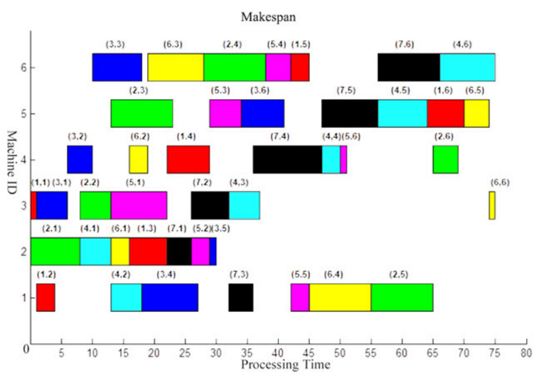

FIGURE 5. Test set 1 scheduling Gantt chart.

Set 2 (λ = 2) is a non-emergency. According to the algorithm flow, the parallel scheduling should be adopted. The experiment results are as follows:

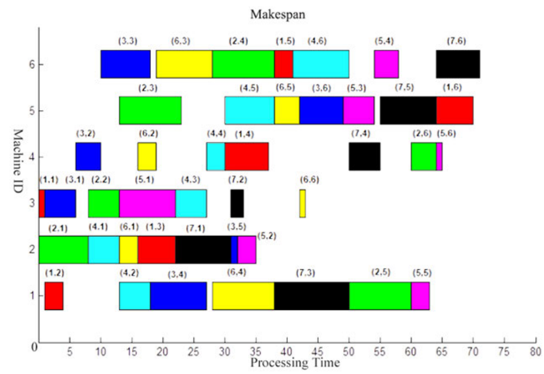

FIGURE 6. Test set 2 scheduling Gantt chart.

TABLE 5. Test set 1 results.

Scheduling method	OTC/OT	NTC /NT	Penalty cost	Result
Parallel scheduling	None	None	None	Parallel priority scheduling
Parallel priority scheduling	75/75	66/68.4	0	

- Test set 3 experiment results

Set 3 (λ = 1.5) is a general emergency. According to the selective DJSP scheduling strategy described in section III, it is necessary to combine the results of parallel scheduling and parallel priority scheduling to select an optimal scheduling method. The experiment results are as follows:

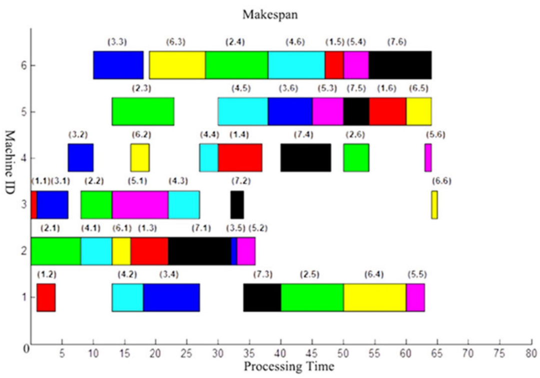

FIGURE 7. Test set 3 scheduling Gantt chart.

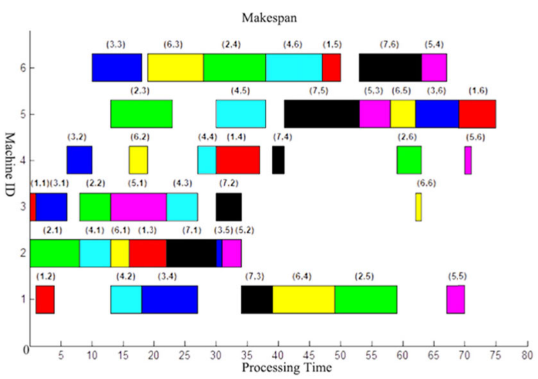

FIGURE 8. Test set 4 scheduling Gantt chart.

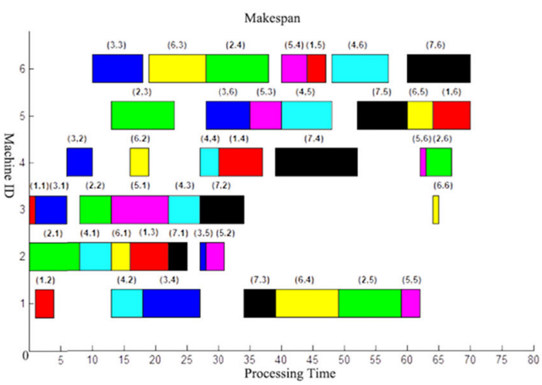

FIGURE 9. Test set 5 scheduling Gantt chart.

TABLE 6. Test set 2 results.

Scheduling method	OTC/OT	NTC /NT	Penalty cost	Result
Parallel scheduling	70/75	71/72.8	0	Parallel scheduling
Parallel priority scheduling	None	None	None	

In order to prove the correctness of the selective scheduling, parallel priority scheduling is adopted for the test set 3, and the experiment results are shown in Table 8. The makespan for parallel scheduling is 65, which is less than

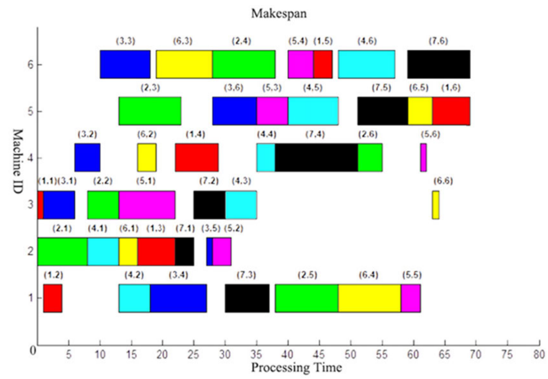

FIGURE 10. Test set 6 scheduling Gantt chart.

TABLE 7. Test set 3 results.

Scheduling method	OTC/OT	NTC /NT	Penalty cost	Result
Parallel scheduling	65/75	64/66	0	Parallel scheduling
Parallel priority scheduling	None	None	None	

TABLE 8. Test set 3 parallel priority scheduling results.

Scheduling method	OTC/OT	NTC /NT	Penalty cost
Parallel priority scheduling	70/75	62/66	0

TABLE 9. Test set 4 results.

Scheduling method	OTC/OT	NTC /NT	Penalty cost	Result
Parallel scheduling	69/75	69/67.15	1.85	Parallel priority scheduling
Parallel priority scheduling	75/75	63/67.15	0	

the makespan of 70 with parallel priority scheduling. Parallel scheduling is better than parallel priority scheduling, so the scheduling method is chosen correctly.

- Test set 4 experiment results

Set 4 ($\lambda = 1.5$) is a general emergency, as same as Set 3 to select an optimal scheduling method. The experiment results are as follows:

From the TABLE 9, there is no penalty cost for parallel priority scheduling, and the workshop should choose this scheduling method. Therefore, parallel scheduling should not be performed inactual scheduling activities.

- Test set 5 experiment results

Set 5 ($\lambda = 1.5$) is a general emergency, as same as Set 3 to select an optimal scheduling method. The experiment results are as follows:

In order to prove the correctness of the selective scheduling, parallel priority scheduling is adopted for the test set 5, and the experiment results are shown in Table 11.

TABLE 10. Test set 5 results.

Scheduling method	OTC/OT	NTC /NT	Penalty cost	Result
Parallel scheduling	70/75	70/72.9	0	Parallel priority scheduling
Parallel priority scheduling	None	None	None	

TABLE 11. Test set 5 parallel priority scheduling results.

Scheduling method	OTC/OT	NTC /NT	Penalty cost
Parallel priority scheduling	77/75	68/72.9	2

TABLE 12. Test set 6 results.

Scheduling method	OTC/OT	NTC /NT	Penalty cost	Result
Parallel scheduling	69/75	69/68.3	0.7	Parallel scheduling
Parallel priority scheduling	76/75	64/68.3	1	

TABLE 13. Solving speed test set 1.

Number	Number of machines	Number of workpieces
1	5	20
2	5	30
3	5	40
4	5	50
5	5	60
6	5	70
7	5	80

Obviously, parallel scheduling is better than parallel priority scheduling, so the scheduling method is chosen correctly.

- Test set 6 experiment results

Set 6 ($\lambda = 1.5$) is a general emergency, as same as Set 3 to select an optimal scheduling method.The experiment results are as follows:

From the TABLE 12, both scheduling methods will generate penalty cost, and the workshop should choose the scheduling method with less penalty cost. Therefore, parallel priority scheduling should not be performed in actual scheduling activities.

C. ALGORITHM OPERATION EFFICIENCY

The algorithm combines events of different urgency and the results of different scheduling methods to select the scheduling method. Because the algorithm calculates up to two scheduling methods in a single time, the running time is longer. It can be found from the experiment results that the number of parallel scheduling is more. Therefore, parallel scheduling is taken as an example to verify the operating efficiency of the algorithm. In the following, some large-scale test sets (Table 13, 14) will be set to test the speed of the

TABLE 14. Solving speed test set 2.

Number	Number of machines	Number of workpieces
1	15	15
2	15	20
3	15	25
4	15	30

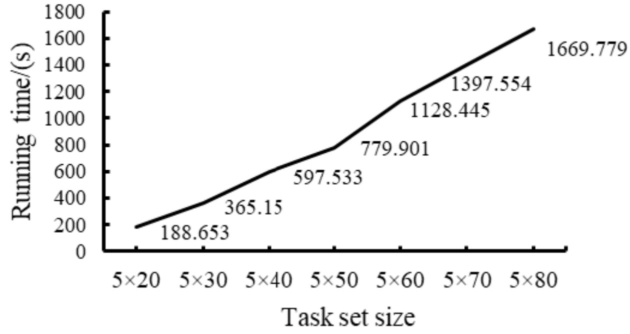

FIGURE 11. Test the runtime curve for set 1.

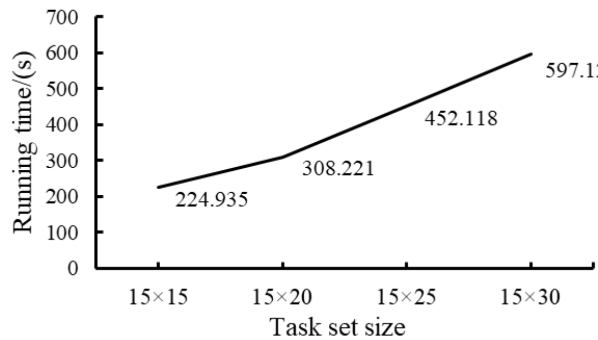

FIGURE 12. Test the runtime curve for set 2.

algorithm for different problem sizes. The results are shown in Figures 11 and 12.

Based on the running time curve of different problem scales, it is not difficult to find that the growth trend of the running time is linear as the scale of the increasing problem. The larger the problem size is, the running time is a little long. Because the ASC has the ability to perform parallel computing, the ACS has an advantage in dealing with large-scale practical problems. In this paper, the selective scheduling strategy designed by ACS has the ability to achieve results in large-scale practical problems.

VI. CONCLUSION

Based on the classification of dynamic events according to the urgency degree, the selective scheduling strategy based on ACS is studied, and the DJSP with the minimum makespan and the minimum penalty cost are solved successfully. The scheduling strategy combines dynamic event of different urgency degree with production efficiency to produce an optimal scheduling strategy. When the urgent orders arrive, it can help the enterprise to choose an optimal scheduling method. Even in the special case where penalty costs are bound to occur, the scheduling method with the minimal penalty costs can be obtained according to the algorithm process of the selective scheduling strategy. The economic benefits of the enterprise can be guaranteed to the maximum extent, and the company's ability to deal with the actual complex scheduling situation can be enhanced. Meanwhile, a feasible scheduling strategy is proposed for the enterprise to handle the scheduling activities of multiple orders.

The experiment results show that ACS can be effectively executed in both parallel scheduling and parallel priority scheduling. For different dynamic events, the selective scheduling strategy can generate the most optimal scheduling method. It can be seen that the selective scheduling strategy improves the flexibility and pertinence of scheduling.

It can be seen from the efficiency experiment that the operation efficiency of the algorithm is not high for large-scale scheduling problems. Therefore, our future work is to explore measures to improve the efficiency of the algorithm, in order to solve the problem that the algorithm runs for a long time in large-scale scheduling problems.

REFERENCES

[1] J. Kuhpfahl and C. Bierwirth, "A study on local search neighborhoods for the job shop scheduling problem with total weighted tardiness objective," *Comput. Oper. Res.*, vol. 66, pp. 44–57, Feb. 2016.

[2] M. R. Garey, D. S. Johnson, and R. Sethi, "The complexity of flowshop and jobshop scheduling," *Math. Oper. Res.*, vol. 1, no. 2, pp. 117–129, May 1976.

[3] Q.-K. Pan, L. Gao, L. Wang, J. Liang, and X.-Y. Li, "Effective heuristics and metaheuristics to minimize total flowtime for the distributed permutation flowshop problem," *Expert Syst. Appl.*, vol. 124, pp. 309–324, Jun. 2019.

[4] F. Zhao, H. Liu, Y. Zhang, W. Ma, and C. Zhang, "A discrete water wave optimization algorithm for no-wait flow shop scheduling problem," *Expert Syst. Appl.*, vol. 91, pp. 347–363, Jan. 2018.

[5] F. Zhao, Y. Liu, Y. Zhang, W. Ma, and C. Zhang, "A hybrid harmony search algorithm with efficient job sequence scheme and variable neighborhood search for the permutation flow shop scheduling problems," *Eng. Appl. Artif. Intell.*, vol. 65, pp. 178–199, Oct. 2017.

[6] O. Engin and A. Güçlü, "A new hybrid ant colony optimization algorithm for solving the no-wait flow shop scheduling problems," *Appl. Soft Comput.*, vol. 72, pp. 166–176, Nov. 2018.

[7] I. A. Chaudhry, "Job shop scheduling problem with alternative machines using genetic algorithms," *J. Cent. South Univ. Technol.*, vol. 19, no. 5, pp. 1322–1333, May 2012.

[8] M. Gen, R. Cheng, and D. Wang, "Genetic algorithms for solving shortest path problems," in *Proc. IEEE Int. Conf. Evol. Comput. (ICEC)*, Apr. 1997, pp. 401–406.

[9] C. Y. Zhang, P. Li, Y. Rao, and Z. Guan, "A very fast TS/SA algorithm for the job shop scheduling problem," *Comput. Oper. Res.*, vol. 35, no. 1, pp. 282–294, Jan. 2008.

[10] N. Q. Jian, X. H. Cheng, Z. Z. Zheng, W. Z. Zhen, and W. J. Feng, "Survey of particle swarm optimization algorithm," *Pattern Recognit. Artif. Intell.*, vol. 40, no. 3, pp. 479–496, 2007.

[11] F. Zhao, Y. Liu, C. Zhang, and J. Wang, "A self-adaptive harmony PSO search algorithm and its performance analysis," *Expert Syst. Appl.*, vol. 42, no. 21, pp. 7436–7455, Nov. 2015.

[12] F. Glover, J. P. Kelly, and M. Laguna, "Genetic algorithms and tabu search: Hybrids for optimization," *Comput. Oper. Res.*, vol. 22, no. 1, pp. 111–134, Jan. 1995.

[13] V. P. Eswaramurthy and A. Tamilarasi, "Hybridizing tabu search with ant colony optimization for solving job shop scheduling problems," *Int. J. Adv. Manuf. Technol.*, vol. 40, nos. 9–10, pp. 1004–1015, Feb. 2009.

[14] F. Zhao, S. Qin, Y. Zhang, W. Ma, C. Zhang, and H. Song, "A two-stage differential biogeography-based optimization algorithm and its performance analysis," *Expert Syst. Appl.*, vol. 115, pp. 329–345, Jan. 2019.

[15] F. Zhao, F. Xue, Y. Zhang, W. Ma, C. Zhang, and H. Song, "A hybrid algorithm based on self-adaptive gravitational search algorithm and differential evolution," *Expert Syst. Appl.*, vol. 113, pp. 515–530, Dec. 2018.

[16] M. Dorigo, "Optimization, learning and natural algorithms," Ph.D. dissertation, Politecnico di Milano, Milan, Italy, 1992.

[17] P. Guo and W. J. Yan, "The review of ant colony algorithm based on TSP," *J. Comput. Sci. China*, vol. 34, no. 10, pp. 181–184, 2007.

[18] X. Qu and C. Zuo, "Improved ant colony algorithm and application in the VRP," *Sci. Technol. Eng.*, vol. 8, no. 2, pp. 563–565, 2008.

[19] B. Bullnheimer, R. F. Hartl, and C. Strauss, "An improved ant system algorithm for the vehicle routing problem," *Ann. Oper. Res.*, vol. 89, pp. 319–328, Jan. 1999.

[20] Z. Z. Wang, J. X. Cheng, H. G. Fang, and F. L. Qian, "A hybrid optimization algorithm solving vehicle routing problems," *Oper. Res. Manage. Sci.*, vol. 13, no. 6, pp. 48–52, 2004.

[21] A. Colorni, M. Dorigo, V. Maniezzo, and M. Trubian, "Ant system for job-shop scheduling," *Oper. Res. Statist. Comput. Sci.*, vol. 34, no. 1, pp. 39–53, 1994.

[22] J. Heinonen and F. Pettersson, "Hybrid ant colony optimization and visibility studies applied to a job-shop scheduling problem," *Appl. Math. Comput.*, vol. 187, no. 2, pp. 989–998, Apr. 2007.

[23] C. Blum and M. Sampels, "An ant colony optimization algorithm for shop scheduling problems," *J. Math. Model. Algorithms*, vol. 3, no. 3, pp. 285–308, 2004.

[24] E. Nowicki and C. Smutnicki, "A fast taboo search algorithm for the job shop problem," *Manage. Sci.*, vol. 42, no. 6, pp. 797–813, Jun. 1996.

[25] M. Dorigo and L. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 53–66, Apr. 1997.

[26] S. C. Graves, "A review of production scheduling," *Oper. Res.*, vol. 29, no. 4, pp. 646–675, 1981.

[27] S. Yan, "Research on flexible job-shop static scheduling problem and algorithm," M.S. thesis, Dept. College Mech. Eng., Chongqing Univ., Chongqing, China, 2013.

[28] W. Zhang, P. Zheng, and X. Wu, "Solution to flexible job shop scheduling problems with capacitated constraints based on ant colony & genetic algorithms," *Comput. Integr. Manuf. Syst.*, vol. 13, no. 2, pp. 333–337, 2007.

[29] X. Huang, X. Zhang, and Y. Ai, "ACO integrated approach for solving flexible job-shop scheduling with multiple process plans," *Comput. Integr. Manuf. Syst.*, vol. 24, no. 3, p. 3, 2018.

[30] J. Mohan, K. Lanka, and A. N. Rao, "A Review of dynamic job shop scheduling techniques," *Digit. Manuf. Transforming Ind. Towards Sustain. Growth*, vol. 30, pp. 34–39, Jan. 2019.

[31] B. Li, C. Wang, and J. Xiong, "Solution method of dynamic job-shop scheduling problem based on adaptive ant colony algorithm," *Mech. Electr. Eng. Mag.*, vol. 26, no. 7, pp. 93–96, 2009.

[32] W. Chen, S. Wang, and H. Huang, "Research on ant colony algorithm for multi-agent based workshop dynamic scheduling," *Modular Mach. Tool Autom. Manuf. Technique*, no. 6, pp. 56–57 and 59, Jun. 2004.

[33] G. Zhang and S. Dang, "Research on dynamic flexible job shop scheduling problem under data driven," *Machinery Des. Manuf.*, no. 6, pp. 267–269, Jun. 2017.

[34] V. Suresh and D. Chaudhuri, "Dynamic scheduling—A survey of research," *Int. J. Prod. Econ.*, vol. 32, no. 1, pp. 53–63, Aug. 1993.

[35] D. Liang, Z. H. Zhan, and J. Zhang, "An adaptive ant colony system for public bicycle scheduling problem," in *Proc. Neural Inf. Process. (ICONIP)*, 2018, pp. 417–429.

[36] J. Xiong, "The method to resolve job-shop scheduling problem based on improved ant colony algorithm," M.S. thesis, Dept. College Info. Eng., Zhejiang Univ. Technol., Hangzhou, China, 2007.

[37] S. Lijun and Z. Hu, "Dynamic flexible job shop scheduling research based on rolling time domain optimization strategy," *Modern Manuf. Eng.*, no. 2, pp. 30–35 and 42, Feb. 2015.

[38] R. J. Abumaizar and J. Svestka, "Rescheduling job shops under random disruptions," *Int. J. Prod. Res.*, vol. 35, no. 7, pp. 2065–2082, Jul. 1997.

[39] K.-L. Huang and C.-J. Liao, "Ant colony optimization combined with taboo search for the job shop scheduling problem," *Comput. Oper. Res.*, vol. 35, no. 4, pp. 1030–1046, Apr. 2008.

[40] Z. W. Ye and Z. B. Zheng, "Configuration of parameters α,β,ρ in ant algorithm," *Geomatics Inf. Sci. Wuhan Univ.*, vol. 29, no. 7, pp. 597–601, 2004.

[41] M. Dorigo, M. Dorigo, V. Manjezzo, and M. Trubian "Ant system for job-shop scheduling," *Oper. Res. Statist. Comput. Sci.*, vol. 34, no. 1, pp. 39–53, 1994.

[42] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: Optimization by a colony of cooperating agents," *IEEE Trans. Syst., Man, Cybern. B. Cybern.*, vol. 26, no. 1, pp. 29–41, Feb. 1996.

[43] F. Alharbi, Y.-C. Tian, M. Tang, W.-Z. Zhang, C. Peng, and M. Fei, "An ant colony system for energy-efficient dynamic virtual machine placement in data centers," *Expert Syst. Appl.*, vol. 120, pp. 228–238, Apr. 2019.

[44] Y. Wang, W. Wang, H. Yu, and L. Chen, "Dynamic balance adaptive colony algorithm solving job-shop scheduling," *Comput. Integr. Manuf. Syst.*, vol. 19, no. 10, pp. 2521–2527, 2013.

YI WAN (Member, IEEE) received the Ph.D. degree in materials science and engineering from the Nanjing University of Science and Technology, Nanjing, in 2008. She is currently an Associate Professor with the School of Environmental Science, Nanjing Xiaozhuang University, Nanjing. Her research interests include analysis of tribological systems, simulation planning and design, and innovation method.

TIAN-YU ZUO is currently pursuing the bachelor's degree with the School of Mechanical and Electrical Engineering, Sanjiang University, Nanjing, China. His major is mechanical engineering and automation. His current research interests include intelligent manufacturing and evolutionary algorithms.

LIN CHEN received the Ph.D. degree in mechanical engineering from Southeast University, Nanjing, in 2014. He is currently the Vice Manager of Jiangsu Shentong Valve Company, Ltd., Nantong. His research interests include automation system design, precise motion control theory, and design of tribological systems.

WEN-CHENG TANG received the B.S. and M.S. degrees in mechanical engineering from Southeast University, Nanjing, in 1987. He has been a Professor with the School of Mechanical Engineering, Southeast University. His research interests include computer integrated manufacturing systems, friction and wear of functional components, and static and dynamic analysis and optimal design of structural parts.

JIN CHEN is currently pursuing the master's degree with the Nanjing University of Aeronautics and Astronautics. His current research interests include artificial intelligence and data mining.