IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 18, 2019, accepted April 7, 2019, date of publication April 16, 2019, date of current version May 2, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2911028

Artificial-Molecule-Based Chemical Reaction Optimization for Flow Shop Scheduling Problem With Deteriorating and Learning Effects

YAPING FU[1], MENGCHU ZHOU[2,3], (Fellow, IEEE), XIWANG GUO[2,4], (Member, IEEE), AND LIANG QI[5], (Member, IEEE)

[1]School of Business, Qingdao University, Qingdao 266071, China
[2]Helen and John C. Hartmann Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102, USA
[3]Renewable Energy Research Group, King Abdulaziz University, Jeddah 21589, Saudi Arabia
[4]Computer and Communication Engineering College, Liaoning Shihua University, Fushun 113001, China
[5]Department of Intelligent Science and Technology, Shandong University of Science and Technology, Qingdao 266590, China

Corresponding author: MengChu Zhou (zhou@njit.edu)

This work was supported in part by the National Natural Science Foundation of China under Grant 61703220 and Grant 71871105.

ABSTRACT Industry 4.0 is widely accepted to guide a novel and promising production paradigm where many advanced intelligent machines and latest technologies are utilized. The self-optimization and self-training behaviors of advanced intelligent machines make them more and more proficient when processing jobs; while the abrasion of their components reduces their work efficiency in the manufacturing process. Therefore, we address a flow shop scheduling problem with deteriorating and learning effects, where the processing time of jobs is a function of their starting time and positions in a schedule. In order to solve it efficiently, an artificial-molecule-based chemical reaction optimization algorithm is proposed. A set of artificial molecules are constructed based on some elitist solutions and adaptively injected into the population, which can enhance and balance exploration and exploitation abilities. The simulation experiments are carried out on a set of stochastic test problems with different sizes. The experimental results show that the proposed algorithm performs better than its peer algorithms in solving the investigated problem.

INDEX TERMS Industry 4.0, deteriorating and learning effects, flow shop scheduling, chemical reaction optimization, artificial molecules.

I. INTRODUCTION

In recent years, many countries and organizations have proposed some novel production paradigms to advance manufacturing processes [1]. Industry 4.0 is one of the most popular and promising production paradigms in advanced manufacturing area, and it has been regarded as a future direction of manufacturing industry [2]. It is viewed as the fourth industrial revolution and applies the principles of cyber-physical systems, Internet of Things, and smart systems with human-machine interaction paradigms [3], [4]. In contrast to conventional manufacturing systems, an obvious change brought by Industry 4.0 is that advanced intelligent machines are widely applied in a manufacturing system. These advanced machines are capable of self-optimization, self-training and self-maintaining by collecting real-time information and optimizing their behaviors [5]. Thus, some new characteristics emerge in this kind of manufacturing systems.

Scheduling problems have received considerable attention because of their importance in both manufacturing systems and optimization research [6]–[14]. In classical scheduling problems, the processing time of jobs is assumed to be constant. This is no longer true partially caused by the growing applications of advanced intelligent machines. Their self-optimization and self-training ability of advanced intelligent machines can make them more and more proficient when processing jobs; while the abrasion and/or degradation of their components can reduce their work efficiency which thus make the processing time of jobs variable in a manufacturing

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

process. In the past few years, many deteriorating and learning models were proposed to describe a schedule problem with variable job processing time. A deteriorating model is proposed in [15] for a single machine scheduling problem. Its actual processing time of jobs is an increasing function of their starting time. A learning model is introduced by Biskup [16] where its actual processing of jobs is assumed to be a log-linear function of its position index. These two pioneering studies [15], [16] result in more suitable scheduling models for real-world applications, e.g., steel making [17], cleaning assignment [18] and emergence medicine [19].

Flow shop scheduling is one of the well-known scheduling problems [20]–[22]. Recently, there have been a growing interest in it by considering deteriorating and learning effects [23]. Lee *et al*. [24] consider a problem with the deteriorating effect and the objective of minimizing the maximum tardiness, and design a branch and bound algorithm for solving it. Cheng *et al*. [25] investigate a two-machine scheduling problem with the deteriorating effect, and design a branch and bound algorithm to minimize its total flow time. Vahedi-Nouri *et al*. [26] consider the learning effect with maintenance activities and the objective of minimizing the total flow time. In order to solve it, a simulating annealing algorithm combined with heuristic rules is developed. Wang *et al*. [27] investigate a two-machine scheduling problem with deteriorating and learning effects, and a branch and bound algorithm is designed to minimize the makespan. Shiau *et al.* [28] propose a two-agent two-machine scheduling problem with the learning effect, and the objective is to minimize the total completion time of an agent subject to the maximum tardiness of the other agent. Liu and Feng [29] consider a no-wait two-machine scheduling problem with the learning effect, and the processing time of a job is dependent on its allocated resource and position index in a schedule. Its objective is to find an optimal sequence of jobs and optimal resource allocation. Wu *et al.* [30] investigate a two-machine problem with learning effect. A branch and bound algorithm with dominated properties is designed to minimize the makespan. Due to the complexity of deteriorating and learning models, most of the existing studies focus on two or three-machine cases. There are few studies that concern a general flow shop scheduling with multiple series machines, which is commonly seen in real-world manufacturing systems. In addition, the market environment has become increasingly competitive, and decision-makers tend to make frequent production changes in order to provide customers with more product varieties. Because of the use of advanced intelligent equipment and technologies, machines have to spend more time to optimize and train themselves to be more proficient for new products. Their efficiency may become lower with time because of the abrasion of their components. Thus, the deteriorating and learning effects appear and should be taken into account simultaneously when making a scheduling decision. Therefore, general flow shop scheduling considering both effects has been one of the most challenging optimization problems. Flow shop scheduling with more than two machines has proven to be an NP-hard problem. In the past years, mathematical programming approaches and metaheuristics are used to solve flow shop scheduling problems. The former needs to take problems' characteristics into account and can find their optimal solutions. Its performance usually depends on problems' complexity and obtained dominated properties of optimization problems. The latter does not depend on the characteristics of optimization problems and can find optimal and near-optimal solutions with an acceptable time. In the investigated problem with both deteriorating and learning effects, the actual processing time of jobs cannot be determined in advance since they depend on the processing sequence of jobs. Thus, they make a flow shop scheduling model more complicated and difficult than a classical flow shop case and often make exact optimal solution methods helpless. Consequently, this study chooses metaheuristics as its optimizer.

Metaheuristic algorithms are usually based on biological intelligence and physical phenomena [31]–[35]. They can be classified into single-solution-based and population-based metaheuristics. For example, simulated annealing algorithm mimics a physical annealing process and it is a single-solution-based metaheuristic. Chemical reaction optimization simulates a molecule reaction process and it belongs to a population-based metaheuristic. Particle swarm optimization and artificial bee colony are swarm intelligence optimization algorithms. Swarm intelligence optimization are usually nature-inspired optimization algorithms which are also called population-based metaheuristics. For instance, artificial bee colony imitates the collecting honey process of bees. Metaheuristics are able to find optimal or near-optimal solutions and can be successfully used to solve complex optimization problems [36]–[42]. In recent years, an efficient chemical reaction optimization (CRO) has been proposed by Lam and Li [43] to deal with combinational optimization problems. CRO simulates four types of chemical reactions via a set of molecules (population) in a closed container. The chemical reactions tend to direct the molecules to the lowest energy state, i.e., the optimal solution of an optimization problem. Recently, CRO has been employed to tackle such optimization problems as grid scheduling [44] and production scheduling [45], [46]. The experimental comparisons demonstrate that CRO is more competitive than the other population-based intelligent optimization algorithms, e.g., tabu search, fast ant system and simulated annealing algorithms [44]–[46]. This work aims to incorporate some novel techniques to balance its exploration and exploitation abilities, which results in a more powerful CRO-based algorithm for complex optimization problems.

By analyzing the existing studies, we conclude that the prior researches mainly focus on two or three-machine flow shop scheduling problems with deteriorating and learning effects. Moreover, there are few studies taking both effects into account simultaneously. In the context of Industry 4.0 framework, it becomes necessary to consider them together because of the use of advanced intelligent machines

and technologies in manufacturing industry. In addition, there are usually multiple series machines in a real production process. Thus, the existing two or three-machine models and algorithms are not suitable for dealing with this problem. To handle it, this work proposes a general Flow-shop Scheduling Problem with Deteriorating and Learning effects (FSP-DL). It has multiple machines and subjects to the traditional constraints, e.g., job precedence constraints.

In addition, a novel artificial-based-molecule chemical reaction optimization algorithm (ACRO) is proposed to deal with the investigated problem. Compared with the existing researches, we have three contributions: (1) A general flow shop scheduling problem with deteriorating and learning effects is proposed, which can better describe some practical production processes. (2) To solve this problem, a novel method called ACRO is designed, where a set of artificial molecules are adaptively injected into the population. It can better balance the exploration and exploitation of the standard CRO, and help to obtain a satisfying schedule decision for decision-makers. (3) Experimental comparison is done to validate the effectiveness and feasibility of the proposed algorithm in coping with this problem.

The reminder of this paper is organized as follows. Section II is the problem definition where the formulation and notations are described, and a mixed integer programming model is presented. Section III describes the basic framework of CRO. The solution method is proposed in Section IV. Simulation experiments are carried out and the results are analyzed in Section V. Section VI gives the conclusions.

II. PROBMENT STATEMENT

The investigated FSP-DL can be described as follows: n jobs need to be processed on m machines following the same route. The normal processing time of jobs without any deterioration and learning is deterministic, while the actual one is dependent on their starting time and positions in a schedule. The objective is to minimize the maximum completion time of all jobs, i.e., makespan. The parameters and decision variables are defined as follows:

INDICES

- i machine index set, $i = \{1, 2, 3, \ldots, m\}$, where m is the number of machines.
- j job index set, $j = \{1, 2, 3, \ldots, n\}$, where n is the number of jobs.
- r position index set, $r = \{1, 2, 3, \ldots, n\}$.

PARAMETER

- α_{ij} deteriorating coefficient of job j on machine i.
- β_{ij} learning coefficient of job j on machine i.
- p_{ij} normal processing time of job j on machine i.
- p'_{ij} actual processing time of job j on machine i.
- $\langle r \rangle$ job index in the r-th position.
- s_{ij} starting time of job j on machine i.
- C_{ij} completion time of job j on machine i.
- C_{max} maximum completion time, i.e., makespan.
- G an infinite positive number.

DECISION VARIABLE

x_{jr} a 0-1 integer decision variable, where $x_{jr} = 1$ if job j is processed in the r-th position; otherwise $x_{jr} = 0$.

In the proposed problem, the actual processing time of jobs is formulated as follows:

$$p'_{ij} = \left(p_{ij} + \alpha_{ij} \cdot S_{ij}\right) \cdot \gamma^{\beta_{ij}}. \tag{1}$$

where $\alpha_{ij} > 0$ is a deteriorating coefficient of job j on machine i. It describes a deteriorating effect that the delay to start a job increases its processing time. $\beta_{ij} < 0$ is a learning coefficient of job j on machine i. It indicates the learning effect that the jobs' actual processing time becomes shorter when they are processed later. In (1), $p_{ij}+\alpha_{ij}\cdot S_{ij}$ indicates that the deteriorating effect causes an increasing trend for the jobs' processing time with their starting time; while $\gamma^{\beta_{ij}}$ shows a decreasing trend related to jobs' positions. Therefore, it can describe both deteriorating and learning effects of a job.

Based on these notations, an optimization model of this problem is built as follows:

$$\min C_{max}. \tag{2}$$

$$s.t.$$

$$\sum_{j=1}^{r} x_{jr} = 1, \quad r = 1, 2, 3, \ldots, n. \tag{3}$$

$$\sum_{r=1}^{n} x_{jr} = 1, \quad j = 1, 2, 3, \ldots, n. \tag{4}$$

$$S_{i,(r)} + P'_{i,(r)} \le S_{i+1,(r)}, \quad i = 1, 2, 3, \ldots, m-1, \quad r = 1, 2, 3, \ldots, n-1. \tag{5}$$

$$S_{i,(r)} + P'_{i,(r)} \le S_{i+1,(r)}, \quad i = 1, 2, 3, \ldots, m, \quad r = 1, 2, 3, \ldots, n. \tag{6}$$

$$p'_{i,(r)} = \left(p_{i,(r)} + \alpha_{i,(r)} \cdot S_{i,(r)}\right) \cdot r^{\beta_{i,(r)}}, \quad i = 1, 2, 3, \ldots, m, \ r = 1, 2, 3, \ldots, n. \tag{7}$$

$$C_{max} \ge C_{mj}, j = 1, 2, 3, \ldots, n. \tag{8}$$

$$S_{ij} \ge 0, C_{ij} \ge 0, \quad i = 1, 2, 3, \ldots, m, \quad j = 1, 2, 3, \ldots, n. \tag{9}$$

$$x_{jr} \in \{0, 1\}, \quad j = 1, 2, 3, \ldots, n, \quad r = 1, 2, 3, \ldots, n. \tag{10}$$

where (2) represents that the objective is to minimize the makespan; (3) ensures that each job can be processed at only one position; (4) stipulates that each position can arrange only one job; (5) defines that each job can be processed on only one machine at any time; (6) ensures that each machine can process only one job at any time; (7) denotes that the actual processing time of a job is a function of its starting time and position index; (8) defines makespan; (9) gives the range of variables; and (10) represents that decision variable x_{jr} can take values of 0 or 1.

III. STANDARD CRO

Chemical reaction is a process of breaking molecules into atoms and rearranging them to form new molecules [47], [48]. CRO was proposed by Lam and Li [43] in 2010. It simulates a chemical reaction of a set of molecules microscopically which tries to capture the energy in a reaction process. Its basic idea is described next.

A. ELEMENTARY ELEMENT OF CRO

A molecule is an elementary element of CRO, and its structure represents a solution of an optimization problem. It is composed of several atoms characterized by atom types, i.e., bond length, angle and torsion. Molecules are distinguished by their atoms and types. In the standard CRO, each molecule has two kinds of energies, i.e., potential and kinetic energies denoted as P and Q respectively. P corresponds to the objective value of a molecule, while Q indicates the ability of a molecule that escapes from a local optimum. Let ω be a molecule. Then its potential energy is obtained by:

$$P_\omega = f(\omega). \tag{11}$$

Suppose that a new molecule ω' is generated from ω. The condition to allow ω' to replace ω is:

$$P_\omega + Q_\omega \geq P_{\omega'}. \tag{12}$$

B. ELEMENTARY REACTION OF CRO

CRO deals with an optimization problem via four elementary reactions, i.e., on-wall ineffective collision, decomposition, inter-molecular ineffective collision and synthesis. A new molecule is generated from some original molecules in population through these reactions. If a given condition is satisfied, a new molecule replaces an original one.

1) On-wall ineffective collision: This occurs if a molecule ω hits a wall and bounces back. It changes ω''s structure, and generates a new molecule ω'. After this collision, the kinetic energy of ω is transferred to an energy buffer, while that of ω' is computed as follows:

$$KE_\omega = \left(PE_\omega + KE_\omega - PE_{\omega'}\right) \times q. \tag{13}$$

where $q \in [K_p, 1]$, and K_p denotes the maximum percentage of kinetic energy loss each time. $1 - q$ represents the percentage of kinetic energy transferred to the energy buffer. The energy in the buffer is updated as follows:

$$B = B + \left(P_\omega + Q_\omega - P_{\omega'}\right) \times (1 - q). \tag{14}$$

where B denotes the total energy stored in the buffer.

2) Decomposition: The decomposition takes place when ω hits a wall. It decomposes ω into two or more new molecules. Two situations are considered: (1) ω has sufficient energy to complete its decomposition; and (2) the energy stored in the buffer is used to complete it. Suppose that molecules ω_1' and ω_2' are generated in this collision, then we have:

Situation 1:

$$P_\omega + Q_\omega \geq P_{\omega_1'} + P_{\omega_2'}. \tag{15}$$

We have:

$$Q_{\omega_1'} = \left(P_\omega + Q_\omega - P_{\omega_1'} + P_{\omega_2'}\right) \times q. \tag{16}$$

$$Q_{\omega_2'} = \left(P_\omega + Q_\omega - P_{\omega_1'} + P_{\omega_2'}\right) \times (1 - q). \tag{17}$$

Situation 2:

$$B + P_\omega + Q_\omega \geq P_{\omega_1'} + P_{\omega_2'}. \tag{18}$$

We have:

$$Q_{\omega_1'} = \left(P_\omega + Q_\omega - P_{\omega_1'} + P_{\omega_2'} + B\right) \times k_1 \times k_2. \tag{19}$$

$$Q_{\omega_2'} = \left(P_\omega + Q_\omega - P_{\omega_1'} + P_{\omega_2'} + B\right) \times k_3 \times k_4. \tag{20}$$

where k_1, k_2, k_3 and k_4 are randomly generated from an interval [0,1]. The buffer is updated as:

$$B = B + P_\omega + Q_\omega - P_{\omega_1'} - Q_{\omega_1'} - P_{\omega_2'} - Q_{\omega_2'}. \tag{21}$$

3) Inter-molecule ineffective collision: This is used to mimic the process of two molecules ω_1 and ω_2 colliding with each other, bouncing away, and yielding new molecules ω_1' and ω_2'.

$$P_{\omega_1} + Q_{\omega_1} + P_{\omega_2} + Q_{\omega_2} \geq P_{\omega_1'} + P_{\omega_2'}. \tag{22}$$

We have:

$$Q_{\omega_1'} = \left(P_{\omega_1} + Q_{\omega_1} + P_{\omega_2} + Q_{\omega_2} - P_{\omega_1'} - P_{\omega_2'}\right) \times k. \tag{23}$$

$$Q_{\omega_2'} = \left(P_{\omega_1} + Q_{\omega_1} + P_{\omega_2} + Q_{\omega_2} - P_{\omega_1'} - P_{\omega_2'}\right) \times (1 - k). \tag{24}$$

where k is generated randomly from an interval [0, 1].

4) Synthesis: This occurs when more than one molecule collide and are combined together. Suppose that ω_1 and ω_2 collide each other and merge into a new one ω'.

$$P_{\omega_1} + Q_{\omega_1} + P_{\omega_2} + Q_{\omega_2} \geq P_{\omega'}. \tag{25}$$

We have:

$$Q_{\omega'} = P_{\omega_1} + Q_{\omega_1} + P_{\omega_2} + Q_{\omega_2} - P_{\omega'}. \tag{26}$$

IV. PROPOSED ACRO

From the above description, decomposition breaks a molecule into two, while synthesis combines two molecules into one. Since they can change the population size, CRO is a variable population-based metaheuristic algorithm. From the preliminary experiment on the investigated problem, we find that the population size decreases quickly in a search process, which makes this algorithm unable to escape from a local optimum. To overcome this issue, some artificial molecules, which are generated based on the elite solutions found, constructed and injected into the population adaptively. This method can make the population explore more regions, and enhance the balance between its exploration and exploitation abilities. The proposed algorithm is described next.

A. MOLECULE STRUCTURE REPRESENTATION

A suitable molecule structure for CRO can increase its ability to find an optimal solution. In this work, the permutation representation is used to denote a feasible and complete solution. In this representation, an integer at each position indicates its corresponding job index. For example, $< 3, 2, 1, 4, 5 >$ is a molecule structure, and it represents that five jobs are processed on the machines as $3 \rightarrow 2 \rightarrow 1 \rightarrow 4 \rightarrow 5$. In this work, S molecules are randomly generated as an initial population.

B. ON-WALL INEFFECTIVE COLLISION

In the standard CRO, the on-wall ineffective collision is to change the selected molecule by rearranging its molecule structure and generate a new one. In order to make a new molecule different from the previous one, *insert* neighborhood and *swap* neighborhood structures [23] are used.

- *Insert*: it is executed as follows: (1) two positions are randomly selected, and (2) the job in the first position is removed and inserted at the second position.
- *Swap*: (1) two positions are randomly selected, and (2) the jobs in the two positions are swapped.

In the on-wall ineffective collision, one of the two neighborhoods is randomly selected and used to generate a new molecule. If (12) is satisfied, this new molecule is allowed to replace the original one.

C. DECOMPOSITION

The decomposition breaks one molecule into two or more in the standard CRO. In the proposed algorithm, two molecules are generated by decomposing a molecule. It is executed as follows: (1) a job block is selected randomly; (2) two different positions are selected randomly, and a job block is randomly chosen and transferred into the two position, respectively. An example is shown in Fig. 1. Two positions h_1 and h_2 are selected, and a job block containing jobs 7 and 5 is chosen, then two new molecules are generated by transferring the job block before h_1 and after h_2, respectively. New molecules are allowed to replace the original one if (15) is satisfied.

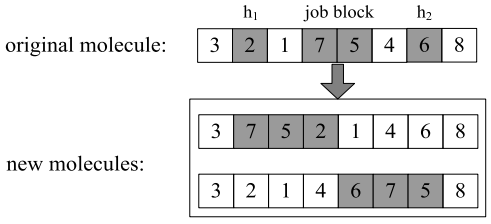


FIGURE 1. Illustration of decomposition reaction.

D. INTER-MOLECULE INEFFECTIVE COLLISION

The inter-molecule ineffective collision occurs when two molecules collide and generate two new molecules. In this work, the collision is realized by an order-based crossover operation [23] between two molecules. Let ω_1 and ω_2 be two molecules which are randomly chosen and ω_1' and ω_2' be two new molecules which are generated by using order-based crossover operation on ω_1 and ω_2. If (22) is satisfied, they are allowed to replace the original ones. An example of this inter-molecule ineffective collision by using order-based crossover operation is given in Fig. 2, where c_1 and c_2 are two randomly generated points. ω_1' stores jobs between c_1 and c_2 in ω_1 as their original positions and the other jobs are from ω_2 as their sequence. ω_2' can be generated by using this approach.

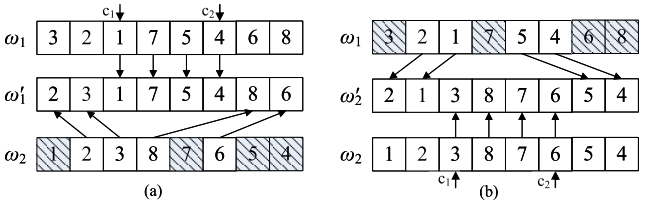


FIGURE 2. Illustration of inter-molecule ineffective collision by using order-based crossover operation.

E. SYNTHESIS

The synthesis is used to combine two or more molecules into one new molecule in the standard CRO. In the proposed algorithm, a molecule is generated by combining two molecules. In this work, the order-based crossover operation in [23] is utilized to generate two molecules, and one of them is randomly selected as a new molecule. The new molecule replaces the original one if (25) is satisfied. Let ω_1 and ω_2 be two molecules which are randomly chosen and ω be a new molecule which are generated by using order-based crossover operation on ω_1 and ω_2. Take the instance in Fig. 2 as an example, we randomly choose a molecule from ω_1' and ω_2' as ω in a synthesis reaction.

F. ARTIFICIAL MOLECULE CONSTRUCTION

In order to increase the exploration and exploitation abilities of CRO, some better molecules, named artificial molecules, are constructed based on the elite solutions that have been found. The artificial molecules are injected into population adaptively to increase its diversity, thus increasing the opportunity for this algorithm to jump out of a local optimum.

In ACRO, a reference solution set is developed to store the better solutions found in a search process. It is updated by the newly generated molecules in each iteration as follows: (1) If the number of molecules in the reference solution set is less than the maximum capacity, then the new molecule ω is directly inserted into the reference solution set, and (2) if the number of molecules in the reference solution set exceeds the maximum capacity, it is inserted after the worst molecule is identified and discarded. Based on the molecules stored in the reference solution set, some artificial molecules are constructed as described in the following steps:

Step 1. Select K elite solutions from the reference solution set based on the roulette wheel selection method [23].

Step 2. Develop a dominated matrix F and probability matrix P based on them. f_{jr}^k is a binary variable defined as:

$$f_{jr}^k = \begin{cases} 1, & \text{if job } j \text{ is in position } r \text{ in the} k\text{thmolecule;} \\ 0, & \text{otherwise.} \end{cases} \quad (27)$$

There are K elite solutions, and the frequency of job j at position r in this dominated matrix denoted by F_{jr} is calculated as:

$$F_{jr} = \sum_{k=1}^{K} f_{jr}^k, \quad j = 1, 2, 3, \ldots, n, \ r = 1, 2, 3, \ldots, n. \quad (28)$$

Step 3. Transform F into a probability matrix. Let p_{jr} be the probability of job j that is assigned to the r-th position. Then p_{jr} is computed as:

$$p_{jr} = \frac{1}{K} F_{jr}, \quad j = 1, 2, 3, \ldots, n, \ r = 1, 2, 3, \ldots, n. \quad (29)$$

Step 4. Construct artificial molecules by using a roulette wheel selection method [23] as shown in Algorithm 1 in Supplementary File. In order to generate an artificial molecule, a random permutation vector of n positions is generated, and the job is assigned to each position by a roulette wheel selection method based on the probability of jobs on this position.

In order to demonstrate the proposed algorithm clearly, the Pseudo-code of ACRO is shown in Algorithm 2 in Supplementary File.

V. EXPERIMENTAL RESULT AND ANALYSIS

In this section, simulation experiments are carried out to evaluate CRO, ACRO and its peers in solving FSP-DL. All the experiments are implemented in C++ on an AMD Athlon 64 Notebook computer running at Dual-Core 1.7 GHz with 1G memory.

A. EXPERIMENTAL SETTING

In order to examine the performance of the proposed algorithm in solving FSP-DL, the standard CRO algorithm (CRO) [43], opposition-based differential evolution algorithm (ODDE) [49] and hybrid modified global-best harmony search algorithm (hmGHS) [50] are chosen as its peer algorithms. CRO is a population-based metaheuristic and it does not employ the artificial molecules in its search process. ODDE and hmGHS belong to evolutionary algorithm [41]. They have been specially developed to deal with flow shop scheduling problems and enjoyed their great performance, and the results show that they outperform differential evolution, particle swarm optimization and genetic algorithm [49], [50], respectively. The design of CRO, ODDE and hmGHS are the same for their original works. When employing CRO to solve the investigated problem, we can roughly analyze its computational complexity in the worst case. The decomposition reaction chooses a job block and inserts it into the other position, and its complexity is $O(n)$. The synthesis reaction uses an order-based crossover operation to generate a new molecule, and its complexity is $O\left(n^2\right)$. In the synthesis reaction, two molecules are generated by using an order-based crossover operation and its complexity is $O\left(n^2\right)$. The on-wall ineffective collision reaction employs swap and insert neighborhoods to generate a new molecule, and thus its complexity is $O(n)$. Since only one reaction is performed at each iteration, the complexity of CRO in the worst case is $O\left(n^2\right)$ at each iteration. The difference between ACRO and CRO is that ACRO uses an approach of injecting artificial molecules. In order to select K elite solutions, its complexity is $O\left(D \cdot \log D\right)$ with a fast sorting approach, where D is the number of solutions in a reference solution set. To construct a probability matrix, we need to calculate each element in it and the complexity is $O\left(K \cdot n^2\right)$. In order to generate a new molecule, each job is chosen based on a roulette wheel selection method, and thus its complexity is $O\left(n^2\right)$. Hence, the complexity of injecting artificial molecules is $O\left(D \cdot \log D + (K+1) \cdot n^2\right)$, and the overall complexity of ACRO in the worst case is $O\left(D \cdot \log D + (K+2) \cdot n^2\right)$ at each iteration. According to the above analysis, ACRO is more complexity than CRO in the worst case. However, the approach of injecting artificial molecules is adaptive and not performed at every iteration. Hence, it could not obviously influence the efficiency of ACRO.

In order to make a fair comparison, all algorithms take the number of fitness evaluations is taken as a stop condition, and the total number of evaluations is set to ρmn, where ρ are set to 50, 100, 200. When there is only one molecule in the population, $\lfloor (1 - g/g_M) \times S \rfloor$ artificial molecules are constructed and injected into the population, where g is the current number of fitness evaluations, g_M is the total number of fitness evaluations, and S is the initial population size. We apply an approach in [54], [55] to generate test problems. This work considers ten test problems with $m \in \{5, 10\}$ and $n \in \{20, 40, 60, 80, 100\}$. For each test problem, the processing time of jobs on each machine is randomly generated from a uniform distribution U[1,30], the deteriorating and learning coefficients are randomly generated from uniform distribution U[0.1, 0.3] and U[-0.3, -0.1], respectively.

For all the test problems, 30 independent replications are made. In each replication, the percentage relative error (ε) is calculated as follows:

$$\varepsilon\,(\mathrm{A}) = \frac{1}{r} \times \sum_{r=1}^{R} \left(\frac{\left(C_i^A - C_i^B\right) \times 100}{C^B} \right). \quad (30)$$

where A $\in$ {CRO, ACRO, ODDE, hm GHS}, C_i^B is the best makespan found by all four algorithms in the i-th run, C_i^A is the makespan obtained by algorithm A, and R= 30 is the total number of independent replications. Obviously, the smaller ε (A), the better algorithm A.

In order to determine an effective parameter combination for ACRO, three key parameters, i.e., population size S, collision rate *collision_rate* (c_r) and the number of elite solutions which is chosen from reference solutions K are determined by using orthogonal experiment. Each of them has four

TABLE 1. Orthogonal experiment design results.

Index	S	c_r	K	ε
1	10	0.2	10	0.1653
2	20	0.2	20	0.2118
3	30	0.2	30	0.1372
4	40	0.2	40	0.1760
5	10	0.4	20	0.1894
6	20	0.4	10	0.1615
7	30	0.4	40	0.1337
8	40	0.4	30	0.1354
9	10	0.6	30	0.1803
10	20	0.6	40	0.1665
11	30	0.6	10	0.1657
12	40	0.6	20	0.1887
13	10	0.8	40	0.2238
14	20	0.8	30	0.1916
15	30	0.8	20	0.1324
16	40	0.8	10	0.1698

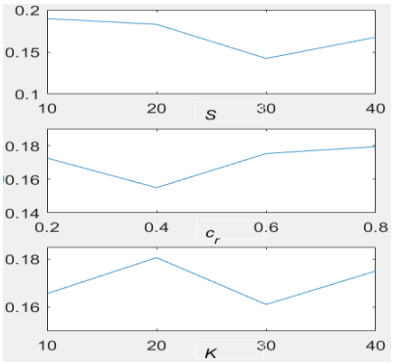


FIGURE 3. Factor level trends.

levels, i.e., $S \in \{10, 20, 30, 40\}$, $c_r \in \{0.2, 0.4, 0.5, 0.8\}$ and $K \in \{10, 20, 30, 40\}$. The other parameters of ACRO are set as follows: Both decomposition and synthesis conditions are set to $10mn$, where m and n represent the numbers of machines and jobs, respectively. The initial kinetic energy for each molecule is set to $100mn$, and the initial energy in the energy buffer is set to $10mn$. The maximum size of a reference solution set is set to 50. ACRO takes the number of fitness evaluations as a stopping termination and it is set to $100mn$ and a test problem 5×40 is taken as an example. The orthogonal experiment design is given in Table 1. For each parameter combination, ACRO runs 30 times independently and the average value ε of 30 times is obtained. The experimental results are given in Table 1. According to them, the trends of each factor level are illustrated in Fig. 3. Then the response value of each parameter is figured out to analyzed the significant rank. The results are given in Table 2. From these experimental results, we can find that S has the most significant impact because it directly influences the algorithm's exploration ability. c_r and K rank the second and third, respectively. c_r controls the number of molecules generated

TABLE 2. Response table and significant rank.

Level	S	c_r	K
1	0.1897	0.1726	0.1656
2	0.1828	0.1550	0.1806
3	0.1423	0.1753	0.1611
4	0.1675	0.1794	0.1750
Delta	0.0474	0.0244	0.0195
Rank	1	2	3

TABLE 3. Comparison of the experimental results via the four algorithms ($\rho = 50$).

$m \times n$	ACRO	CRO	t	ODDE	t	hmGHS	t
5×20	0.0215	0.0402	~	0.2092	$s+$	0.0951	$s+$
5×40	0.0999	0.2170	$s+$	0.1629	$s+$	0.1089	~
5×60	0.0286	0.1123	$s+$	0.1198	$s+$	0.1221	$s+$
5×80	0.0538	0.1553	$s+$	0.2913	$s+$	0.2452	$s+$
5×100	0.0704	0.1941	$s+$	0.3119	$s+$	0.2725	$s+$
10×20	0.0401	0.1520	$s+$	0.2108	$s+$	0.0491	~
10×40	0.0636	0.0954	~	0.1821	$s+$	0.0913	~
10×60	0.0715	0.1767	$s+$	0.1593	$s+$	0.1301	$s+$
10×80	0.1392	0.2104	$s+$	0.3232	$s+$	0.2528	$s+$
10×100	0.1343	0.3084	$s+$	0.2957	$s+$	0.2668	$s+$
mean	0.0723	0.1662		0.2266		0.1634	

by different reactions and thus it can balance global and local searches. K is the number of the selected elite molecules for constructing a probability matrix, which can refine these promising regions by generating artificial molecules. According to the analysis, the suggest parameter combination can be determined as follows: $S = 30$, $c_r = 0.4$ and $K = 30$. Since ACRO is an extended version of CRO, the parameter setting of CRO is the same with ACRO.

The parameter setting of ODDE and hmGHS refers to the work [49] and [50], respectively. They are specially designed for flow shop scheduling problems, which are different from the flow shop scheduling problem with deteriorating and learning effects. Therefore, we need to make orthogonal experiments based on their parameter setting in [49] and [50] to determine their best parameter combination to solve the considered problem. The experimental results are given in Tables S1-S4 in Supplymentary File. Thus, the parameter setting of ODDE is as follows: the population size is set to $2n$, and the mutation probability is 0.70. The parameter setting of hmGHS is given as follows: the memory size is set to 5, the random selection probability is set to 0.95, and the pitch adjustment probability is set to 0.99.

B. EXPERIMENTAL RESULTS AND ANALYSIS

In the following experiments, we compare ACRO with its peers in solving different test problems and t-test in [48], [49] is used to analyze the experimental results. In the following experiments, symbols 's+', 's-' or '~' denote the statistical

TABLE 4. Comparison of the experimental results via the four algorithms ($\rho = 100$).

$m \times n$	ACRO	CRO	t	ODDE	t	hmGHS	t
5×20	0.0239	0.0524	$s+$	0.1698	$s+$	0.0964	$s+$
5×40	0.0360	0.1176	$s+$	0.1054	$s+$	0.0703	$\sim$
5×60	0.0301	0.0940	$\sim$	0.1337	$s+$	0.1181	$s+$
5×80	0.0937	0.1801	$s+$	0.1772	$s+$	0.2141	$s+$
5×100	0.0616	0.1892	$s+$	0.3216	$s+$	0.2246	$s+$
10×20	0.0371	0.1070	$s+$	0.1560	$s+$	0.0374	$\sim$
10×40	0.1101	0.1132	$s+$	0.1938	$s+$	0.0996	$\sim$
10×60	00641	0.1764	$s+$	0.1624	$s+$	0.1350	$s+$
10×80	0.1347	0.2376	$\sim$	0.3711	$s+$	0.2194	$s+$
10×100	0.1220	0.2631	$s+$	0.2635	$s+$	0.2917	$s+$
mean	0.0713	0.1520		0.2054		0.1507	

TABLE 5. Comparison of the experimental results via the four algorithms ($\rho = 200$).

$m \times n$	ACRO	CRO	t	ODDE	t	hmGHS	t
5×20	0.0220	0.0786	$s+$	0.0955	$s+$	0.0736	$s+$
5×40	0.0518	0.1385	$s+$	0.1385	$s+$	0.1381	$s+$
5×60	0.0317	0.1277	$s+$	0.1201	$\sim$	0.1198	$s+$
5×80	0.1043	0.2034	$s+$	0.1741	$s+$	0.2036	$s+$
5×100	0.0847	0.2098	$s+$	0.2933	$s+$	0.2843	$s+$
10×20	0.0224	0.1351	$s+$	0.1680	$s+$	0.0332	$\sim$
10×40	0.1071	0.1872	$s+$	0.2035	$s+$	0.0940	$\sim$
10×60	0.0609	0.1615	$s+$	0.1566	$s+$	0.1694	$s+$
10×80	0.1069	0.2066	$\sim$	0.2810	$s+$	0.1953	$s+$
10×100	0.1199	0.2414	$s+$	0.2590	$s+$	0.2306	$s+$
mean	0.0719	0.1783		0.1890		0.1545	

results of comparing two algorithms by the one-tailed t-test (t) with 58 degree of freedom at a 0.05 level of significance. The t-test results are respectively drawn as 's+', 's-' or '$\sim$' when ACRO is significantly better than, significantly worse than, or statistically equivalent to its peer algorithms. The results with the different number of fitness evaluations are shown in Tables 3-5, respectively.

It can be seen from the experimental results in Table 3 where the least number of fitness evaluations $50mn$ is used, ACRO is the best performer among four algorithms. For all the test problems, ACRO has the best ε, and its overall mean value is equal to 0.0723, which is substantial lower than that of CRO (0.1662), ODDE (0.2266) and hmGHS (0.1634). From the results, we can find that ACRO performs the best in all the test problems. By observing the results via t-test, we can find that ACRO has significantly better than CRO in 8 test problems, and for the rest, the two algorithms are statistical equivalent. Compared to ODDE, ACRO performs significantly better in all the 10 test problems. ACRO outperforms hmGHS significantly in 7 test problems, and the two algorithms performs equivalently in the rest. In addition, we can also find that CRO outperforms ODDE and hmGHS in 2 test problems, while hmGHS shows better performance in the rest. By analyzing the above experimental results, it can be found that ACRO performs better by using the least number of fitness evaluations.

The experimental results of the algorithms with the number of fitness evaluations $100mn$ are shown in Table 4, which are also in favor of ACRO. It can be seen that ACRO has the smallest ε in 9 test problems, and it produces the smallest mean value equal to 0.0713, which is much smaller than that of CRO (0.1520), ODDE (0.2054) and hmGHS (0.1507). The results of t-test show that ACRO is significantly better than CRO in 8 test problems, and statistically equivalent in 2 test problems. Compared to ODDE, ACRO performs significantly better in all the test problems. It can be seen that ACRO outperforms hmGHS significantly in 6 test problems, and equivalently in the rest. Additionally, it can be found that CRO outperforms ODDE and hmGHS in 4 test problems, hmGHS performs better in 5 test problems, and ODDE shows better performance in a test problem. From these experimental results, we can find that ACRO has better performance when the number of fitness evaluations is set to $100mn$.

From the experimental results of the algorithms with the number of fitness evaluations $200mn$ are shown in Table 5. It can be found that ACRO has the best ε in 9 test problems, and its overall mean value is equal to 0.0719, which is lower than that of CRO (0.1783), ODDE (0.1890) and hmGHS (0.1545). The experimental results on t-test indicate that ACRO performs significantly better than CRO in 9 test problems. It is worthy noting that ACRO significantly outperforms ODDE in all the test problems. Compared with hmGHS, ACRO is significantly better in 8 test problems. By comparing CRO, ODDE and hmGHS, we can find that hmGHS outperforms CRO and ODDE in 7 test problems, ODDE performs better in 2 test problems, and CRO shows better performance in the rest. From these experimental results, it can be found that ACRO is a better solver for the investigated problem when the number of fitness evaluations is set to $200mn$.

The experimental results demonstrate that ACRO can achieve the best results among the four algorithms for most of the test problems. Therefore, we can draw a conclusion that it is an efficient optimizer in dealing with the investigated problem. CRO is good at solving discrete combinational optimization problems, and the four types of chemical reactions realize the exploration via global search and exploitation via local search. In coping with this problem, CRO has the superiority in some test problems over ODDE and hmGHS. The main difference between CRO and ACRO is that ACRO applies an approach of injecting artificial molecules to balance exploration and exploitation abilities. From the comparison results between ACRO and CRO when using a different number of fitness evaluations, ACRO performs better than CRO in solving this problem, and thus ACRO has improved CRO's performance. Therefore, we conclude that artificial molecules do increase the diversity of population, avoid early trapping into a local optimum, and enhance the algorithm's

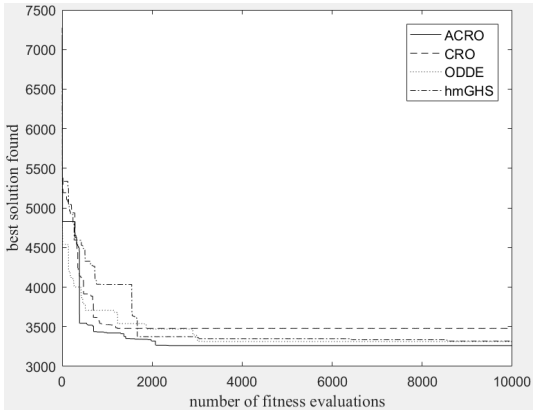


FIGURE 4. The convergence curve of the four algorithms ($\rho = 50$) on the test problem 10 × 20.

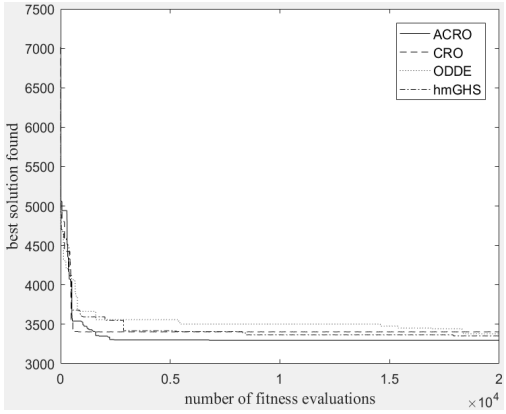


FIGURE 5. 5. The convergence curve of the four algorithms ($\rho = 100$) on the test problem 10 × 20.

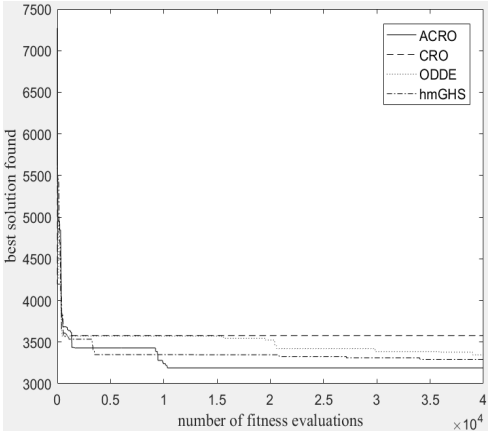


FIGURE 6. The convergence curve of the four algorithms ($\rho = 200$) on the test problem 10 × 20.

local exploitation capacity, which provides a good trade-off between exploration and exploitation.

In order to show their performance in dealing with the investigated problem clearly, the convergence curves of the four algorithms with different stop conditions for the small size, medium size and large size test problems are shown in Figs. 4-6, respectively. They represent the small size (10×20), medium size (10×60) and large size (10×100) test problems, where the first and second numbers represent the numbers of machines and jobs, respectively. From these figures, it can be found that ACRO, on one hand, can converge more quickly than its peer algorithms, and on the other hand, can achieve the better solution than its peer algorithms. Additional cases about medium size and large size are given in Figs. S1-S3 in Supplementary File.

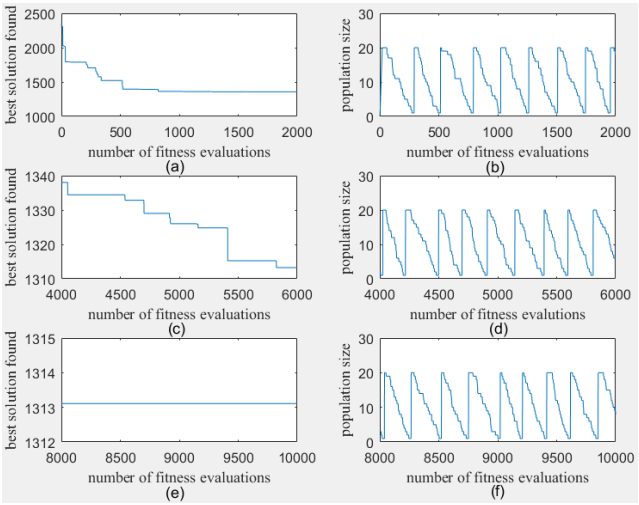


FIGURE 7. Illustration of the change of the best solution and population size in search process.

In the proposed algorithm, the adaptive insertion of artificial molecules is developed to make CRO more powerful in dealing with the investigated problem. In order to show its effectiveness, a test problem is chosen from the above test problem set. The size of this test problem is $m = 5$, $n = 20$ and $\rho = 100$, and the parameter setting of ACRO is the same with that in the above experiment. We utilize ACRO to solve this test problem and record the change of best solutions and population size in its search process. To illustrate the experimental results, a set of graphs is drawn in Fig. 7. Since the number of fitness evaluations is taken as a stop condition in ACRO, we choose three stages from its search process to show the experimental results, i.e., 0-2000, 4000-6000 and 8000-10000, respectively. In Fig. 5, graphs (a), (c) and (e) denote the best solution found in each number of fitness evaluations, and graphs (b), (d) and (f) show the change of their corresponding population size. From Fig.5(a), we can find that the best solution found does not change when $g = 220$, and thus it falls into a local optimum. Nevertheless, the best solution becomes smaller when g is a little bit more than 300. The reason is that a set of artificial molecules is injected into the population which can observe from Fig. 5(b) where the population size becomes large at the same point. It means that the algorithm jumps out of a local optimum because of injecting artificial molecules. The same experimental result can be found from Fig. 5(c)-(d), when g is slightly more than 5400. In Fig. 5(e)-(f), although artificial molecules are injected into the population, the best solution found remains unchanged indicating that the algorithm have been convergent. From the analysis of the experimental results in Fig. 7, we can draw

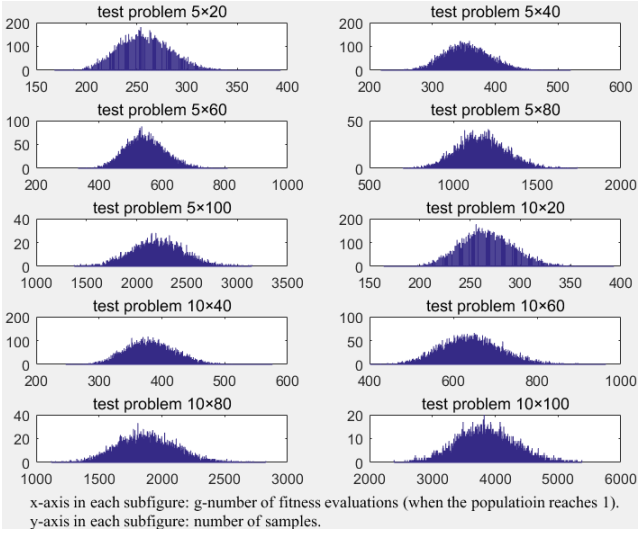


FIGURE 8. Distributions of the number of samples reaching a population of 1 for each test problem.

a conclusion that the adaptive injection of artificial molecules can allow ACRO jump out of a local optimum, which makes it more powerful in coping with the investigated problem.

From the above description, CRO converges very fast as its population size decreases quickly in the search process. In the following experiment, we examine its convergence, and the empirical statistical approach is chosen as an analysis tool since it has been widely applied to examine and analyze the experimental results in the evolutionary computing area [51]–[57]. We utilize CRO to perform 10000 replications on each of the test problems. The test problem sizes are indicated on Fig. 8. In CRO, there is no artificial molecules that are injected into its population in the search process. Thus, this experiment is carried out without introduction of artificial molecules. CRO stops when its population reaches 1, and the number of fitness evaluations that it uses is recorded. The histogram of all the experimental results are shown in Fig. 8. The vertical axis is the number of samples in the experiment, while the number of fitness evaluations when the population reaches 1 is shown on the horizontal axis. We utilize the KS-test [58] to examine this ten sample sets, and the results show that they statistically obey a normal distribution under the significance level at 0.05. If the ten sample sets are transformed to the standard normal distribution respectively, we can draw a conclusion based on the 3σ rule that a sample value which is randomly generated is smaller than 3 times of the deviation at a confidence level 0.0023. Therefore, we can conclude that the number of fitness evaluations when the population size reduces to 1 is smaller than the summation of the mean value and 3 times of the deviation at a confidence level at 0.0023. In this experiment, the number of fitness evaluations for all the test problems when there is only one molecule in population is much smaller than their corresponding stop condition ρmn. Thus, the population size of CRO decreases quickly during the search process, increasing the likelihood of convergence to a local optimum.

Combined with the above experimental results in Tables 3-5 and Figs. 4-6, it can be found that the proposed adaptive insertion of artificial molecules enables CRO to jump out of a local optimum, and thus strengthens its exploration ability. We can conclude that ACRO is a better optimizer for solving the investigated problem. In ACRO, inter-molecule effective collision and synthesis reactions use crossover operation to generate new molecules, which enhances the algorithm's global search ability. Neighborhood search approach is used in the on-wall ineffective collision and decomposition reactions, which strengths the algorithms' local search ability. In addition, an injecting artificial molecule approach is used to avoid falling into local optimum. This approach injects some better molecules into population to increase its diversity, which can enhance the algorithm's exploration ability. The design of ACRO balances the exploration and exploitation abilities, and then it shows better performance in solving the investigated problem.

VI. CONCLUSION

This work addresses flow shop scheduling problems with deteriorating and learning effects (FSP-DL) in the context of Industry 4.0. A mixed integer programming model with the objective of minimizing the makespan is built. The artificial-molecule-based chemical reaction optimization algorithm (ACRO) is proposed to deal with it. The proposed algorithm can solve the model effectively and efficiently, and represents a new method to find a best scheduling decision. The results can guide a decision-maker in making better decisions when both deteriorating and learning effects are considered in a general flow shop environment.

Although the effectiveness of the proposed model and algorithm has been verified, there are still some interesting directions for further research. For example, in an actual production process, uncertain events, e.g., machine breakdown and new order arrival, usually occur because of the difficulty of knowing exact information of jobs and facilities in advance. Additionally, decision-makers have to consider and balance multiple criteria such as production cost and processing quality to find a trade-off solution. Thus, it is necessary to establish new models in uncertain environment [59]–[61] for multi-objective cases [8], [62], [63]. It is also interesting to design some mathematical optimization algorithms, e.g., branch and bound approach, dynamic programming method, to find an optimal solution of the considered problem by incorporating its dominated and optimal properties.

REFERENCES

[1] Y. Fu, J. Ding, H. Wang, and J. Wang, "Two-objective stochastic flow-shop scheduling with deteriorating and learning effect in industry 4.0-based manufacturing system," *Appl. Soft Comput.*, vol. 68, pp. 847–855, Jul. 2018.

[2] M. Prause and J. Weigand, "Industry 4.0 and object-oriented development: Incremental and architectural change," *J. Technol. Manage. Innov.*, vol. 11, no. 2, pp. 104–110, 2016.

[3] D. Gorecky, M. Schmitt, M. Loskyll, and D. Zühlke, "Human-machine-interaction in the industry 4.0 era," in *Proc. 12th IEEE Int. Conf. Ind. Inform. (INDIN)*, Jul. 2014, pp. 289–294.

[4] X. Yue, H. Cai, H. Yan, C. Zou, and K. Zhou, "Cloud-assisted industrial cyber-physical systems: An insight," *Microprocessors Microsyst.*, vol. 39, no. 8, pp. 1262–1270, 2015.

[5] R. Schmidt, M. Möhring, R. C. Härting, C. Reichstein, P. Neumaier, and P. Jozinović, "Industry 4.0—Potentials for creating smart products: Empirical research results," in *Proc. Int. Conf. Bus. Inf. Syst.*, 2015, pp. 16–27.

[6] Y. Fu, G. Tian, Z. Li, and Z. Wang, "Parallel machine scheduling with dynamic resource allocation via a master-slave genetic algorithm," *IEEJ. Trans. Elect. Electron. Eng.*, vol. 13, no. 5, pp. 748–756, 2018.

[7] Y. Hou, N. Wu, M. Zhou, and Z. Li, "Pareto-optimization for scheduling of crude oil operations in refinery via genetic algorithm," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 47, no. 3, pp. 517–530, Mar. 2017.

[8] X. Zuo, C. Chen, W. Tan, and M. Zhou, "Vehicle scheduling of an urban bus line via an improved multiobjective genetic algorithm," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 1030–1041, Apr. 2015.

[9] J. Cheng, F. Chu, and M. C. Zhou, "An improved model for parallel machine scheduling under time-of-use electricity price," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 2, pp. 896–899, Apr. 2018. doi: 10.1109/TASE.2016.2631491.

[10] Y. Yin, S.-R. Cheng, T. C. E. Cheng, D.-J. Wang, and C.-C. Wu, "Just-in-time scheduling with two competing agents on unrelated parallel machines," *Omega*, vol. 63, pp. 41–47, Sep. 2016.

[11] Y. Yin, Y. Yang, D. Wang, T. C. E. Cheng, and C.-C. Wu, "Integrated production, inventory, and batch delivery scheduling with due date assignment and two competing agents," *Nav. Res. Logistics*, vol. 65, pp. 393–409, Aug. 2018.

[12] R. Ruiz, Q.-K. Pan, and B. Naderi, "Iterated Greedy methods for the distributed permutation flowshop scheduling problem," *Omega*, vol. 83, pp. 213–222, Mar. 2019.

[13] K. Peng, Q.-K. Pan, L. Gao, X. Li, S. Das, and B. Zhang, "A multi-start variable neighbourhood descent algorithm for hybrid flowshop rescheduling," *Swarm Evol. Comput.*, vol. 45, pp. 92–112, Mar. 2019.

[14] 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.

[15] J. N. D. Gupta and S. K. Gupta, "Single facility scheduling with nonlinear processing times," *Comput. Ind. Eng.*, vol. 14, no. 4, pp. 387–393, 1988.

[16] D. Biskup, "Single-machine scheduling with learning considerations," *Eur. J. Oper. Res.*, vol. 115, no. 1, pp. 173–178, 1999.

[17] H. Wang, Y. Fu, M. Huang, and J. Wang, "Multiobjective optimisation design for enterprise system operation in the case of scheduling problem with deteriorating jobs," *Enterprise Inf. Syst.*, vol. 10, no. 3, pp. 268–285, 2016.

[18] X.-Y. Wang and J.-J. Wang, "Scheduling deteriorating jobs with a learning effect on unrelated parallel machines," *Appl. Math. Model.*, vol. 38, nos. 21–22, pp. 5231–5238, 2014.

[19] Y. Yin, Y. Wang, T. C. E. Cheng, W. Liu, and J. Li, "Parallel-machine scheduling of deteriorating jobs with potential machine disruptions," *Omega*, vol. 69, pp. 17–28, Jun. 2017.

[20] Y. Fu, H. Wang, and M. Huang, "Integrated scheduling for a distributed manufacturing system: A stochastic multi-objective model," *Enterprise Inf. Syst.*, vol. 13, no. 4, pp. 557–573, 2018. doi: 10.1080/17517575.2018.1545160.

[21] I. Ribas, R. Companys, and X. Tort-Martorell, "An iterated greedy algorithm for solving the total tardiness parallel blocking flow shop scheduling problem," *Expert Syst. Appl.*, vol. 121, pp. 347–361, May 2019.

[22] C.-Y. Cheng, K.-C. Ying, H.-H. Chen, and H.-S. Lu, "Minimising makespan in distributed mixed no-idle flowshops," *Int. J. Prod. Res.*, vol. 57, no. 1, pp. 48–60, 2019.

[23] Y. Fu, H. Wang, G. Tian, Z. Li, and H. Hu, "Two-agent stochastic flow shop deteriorating scheduling via a hybrid multi-objective evolutionary algorithm," *J. Intell. Manuf.*, Jan. 2018, pp. 1–16. doi: 10.1007/s10845-017-1385-4.

[24] W.-C. Lee, W.-C. Yeh, and Y.-H. Chung, "Total tardiness minimization in permutation flowshop with deterioration consideration," *Appl. Math. Model.*, vol. 38, no. 13, pp. 3081–3092, 2014.

[25] M. Cheng, P. R. Tadikamalla, J. Shang, and S. Zhang, "Bicriteria hierarchical optimization of two-machine flow shop scheduling problem with time-dependent deteriorating jobs," *Eur. J. Oper. Res.*, vol. 234, no. 3, pp. 650–657, 2014.

[26] B. Vahedi-Nouri, P. Fattahi, R. Tavakkoli-Moghaddam, and R. Ramezanian, "A general flow shop scheduling problem with consideration of position-based learning effect and multiple availability constraints," *Int. J. Adv. Manuf. Technol.*, vol. 73, nos. 5–8, pp. 601–611, 2014.

[27] J.-B. Wang, P. Ji, T. C. E. Cheng, and D. Wang, "Minimizing makespan in a two-machine flow shop with effects of deterioration and learning," *Optim. Lett.*, vol. 6, no. 7, pp. 1393–1409, 2002.

[28] Y.-R. Shiau, M.-S. Tsai, W.-C. Lee, and T. C. E. Cheng, "Two-agent two-machine flowshop scheduling with learning effects to minimize the total completion time," *Comput. Ind. Eng.*, vol. 87, pp. 580–589, Sep. 2015.

[29] Y. Liu and Z. Feng, "Two-machine no-wait flowshop scheduling with learning effect and convex resource-dependent processing times," *Comput. Ind. Eng.*, vol. 75, pp. 170–175, Sep. 2014.

[30] W.-H. Wu, W.-H. Wu, J.-C. Chen, W.-C. Lin, J. Wu, and C.-C. Wu, "A heuristic-based genetic algorithm for the two-machine flowshop scheduling with learning consideration," *J. Manuf. Syst.*, vol. 35, pp. 223–233, Apr. 2015.

[31] K. Xing, L. Han, M. C. Zhou, and F. Wang, "Deadlock-free genetic scheduling algorithm for automated manufacturing systems based on deadlock control policy," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 3, pp. 603–615, Jun. 2012.

[32] C. Yang, J. Ding, Y. Jin, C. Wang, and T. Chai, "Multitasking multiobjective evolutionary operational indices optimization of beneficiation processes," *IEEE Trans. Autom. Sci. Eng.*, to be published.

[33] J. Ding, C. Yang, Y. Jin, and T. Chai, "Generalized multitasking for evolutionary optimization of expensive problems," *IEEE Trans. Evol. Comput.*, vol. 23, no. 1, pp. 44–58, Feb. 2019.

[34] Z. Fuqing, Q. Shuo, Y. Guoqiang, M. Weimin, Z. Chuck, and S. Houbin, "A factorial based particle swarm optimization with a population adaptation mechanism for the no-wait flow shop scheduling problem with the makespan objective," *Expert Syst. Appl.*, vol. 126, pp. 41–53, Jul. 2019.

[35] Z. Fuqing, Q. Shuo, Z. Yi, M. Weimin, Z. Chuck, and S. Houbin, "A hybrid biogeography-based optimization with variable neighborhood search mechanism for no-wait flow shop scheduling problem," *Expert Syst. Appl.*, vol. 126, pp. 321–339, 2019. 10.1016/j.eswa.2019.02.023.

[36] G. Tian, M. Zhou, and P. Li, "Disassembly sequence planning considering fuzzy component quality and varying operational cost," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 2, pp. 748–760, Apr. 2018.

[37] X. Guo, S. Liu, G. Tian, and M. C. Zhou, "Disassembly sequence optimization for large-scale products with multiresource constraints using scatter search and petri nets," *IEEE Trans. Cybern.*, vol. 46, no. 11, pp. 2435–2446, Nov. 2016.

[38] X. Guo, S. Liu, M. Zhou, and G. Tian, "Dual-objective program and scatter search for the optimization of disassembly sequences subject to multiresource constraints," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 3, pp. 1091–1103, Jul. 2018.

[39] J. Li, Q. Sun, M. Zhou, and X. Dai, "A new multiple traveling salesman problem and its genetic algorithm-based solution," in *Proc. IEEE Int. Conf. Syst. Man, Cybern.*, Oct. 2013, pp. 627–632.

[40] Q. Kang, M. Zhou, J. An, and Q. Wu, "Swarm intelligence approaches to optimal power flow problem with distributed generator failures in power networks," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 2, pp. 343–353, Apr. 2013.

[41] Q.-K. Pan, L. Wang, H.-Y. Sang, J.-Q. Li, and M. Liu, "A high performing memetic algorithm for the flowshop scheduling problem with blocking," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 3, pp. 741–756, Jul. 2013.

[42] L. Wang, S. Wang, and X. Zheng, "A hybrid estimation of distribution algorithm for unrelated parallel machine scheduling with sequence-dependent setup times," *IEEE/CAA J. Autom. Sinica*, vol. 3, no. 3, pp. 235–246, Jul. 2016.

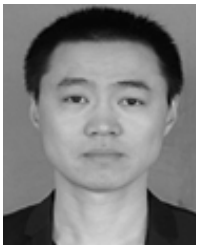
[43] A. Y. S. Lam and V. O. K. Li, "Chemical-reaction-inspired metaheuristic for optimization," *IEEE Trans. Evolut. Comput.*, vol. 14, no. 3, pp. 381–399, Jun. 2010.

[44] J. Xu, A. Y. S. Lam, and V. O. K. Li, "Chemical reaction optimization for task scheduling in grid computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 22, no. 10, pp. 1624–1631, Oct. 2011.

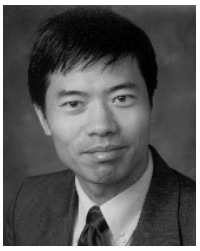
[45] H. Bargaoui, O. B. Driss, and K. Ghédira, "Minimizing makespan in multi-factory flow shop problem using a chemical reaction metaheuristic," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2016, pp. 2919–2926.

[46] H. Bargaoui, O. B. Driss, and K. Ghédira, "A novel chemical reaction optimization for the distributed permutation flowshop scheduling problem with makespan criterion," *Comput. Ind. Eng.*, vol. 111, pp. 239–250, Sep. 2017.

[47] K. Wang *et al.*, "Electrodeposition synthesis of PANI/MnO2/graphene composite materials and its electrochemical performance," *Int. J. Electrochem. Sci.*, vol. 12, pp. 8306–8314, 2017.
[48] K. Wang, J. Pang, L. Li, S. Zhou, Y. Li, and T. Zhang, "Synthesis of hydrophobic carbon nanotubes/reduced graphene oxide composite films by flash light irradiation," *Frontiers Chem. Sci. Eng.*, vol. 12, no. 3, pp. 376–382, 2018.
[49] X. Li and M. Yin, "An opposition-based differential evolution algorithm for permutation flow shop scheduling based on diversity measure," *Adv. Eng. Softw.*, vol. 55, pp. 10–31, Jan. 2013.
[50] L. Wang, Q.-K. Pan, and M. F. Tasgetiren, "A hybrid harmony search algorithm for the blocking permutation flow shop scheduling problem," *Comput. Ind. Eng.*, vol. 61, no. 1, pp. 76–83, 2011.
[51] D. Weyland, "A critical analysis of the harmony search algorithm–how not to solve sudoku," *Oper. Res. Perspect.*, vol. 2, pp. 97–105, Dec. 2015.
[52] T. Zheng and M. Yamashiro, "Solving flow shop scheduling problems by quantum differential evolutionary algorithm," *Int. J. Adv. Manuf. Technol.*, vol. 49, nos. 5–8, pp. 643–662, 2010.
[53] L. Wang, L. Zhang, and D.-Z. Zheng, "An effective hybrid genetic algorithm for flow shop scheduling with limited buffers," *Comput. Oper. Res.*, vol. 33, no. 10, pp. 2960–2971, 2006.
[54] M. Bank, S. M. T. F. Ghomi, F. Jolai, and J. Behnamian, "Application of particle swarm optimization and simulated annealing algorithms in flow shop scheduling problem under linear deterioration," *Adv. Eng. Softw.*, vol. 47, no. 1, pp. 1–6, 2012.
[55] J.-B. Wang and M.-Z. Wang, "Solution algorithms for the total weighted completion time minimization flow shop scheduling with decreasing linear deterioration," *Int. J. Adv. Manuf. Technol.*, vol. 67, nos. 1–4, pp. 243–253, 2013.
[56] Y. Fu, H. Wang, M. Huang, and J. Wang, "A decomposition based multiobjective genetic algorithm with adaptive multipopulation strategy for flowshop scheduling problem," *Natural Comput.*, Feb. 2017, pp. 1–12. doi: 10.1007/s11047-016-9602-1
[57] Y. Fu, H. Wang, M. Huang, J. Ding, and G. Tian, "Multiobjective flow shop deteriorating scheduling problem via an adaptive multipopulation genetic algorithm," *Proc. Inst. Mech. Eng. B, J. Eng. Manuf.*, vol. 232, no. 14, pp. 2641–2650, 2018. doi: 10.1177/0954405417691553.
[58] F. J. Massey, Jr., "The Kolmogorov-Smirnov test for goodness of fit," *J. Amer. Statist. Assoc.*, vol. 46, no. 253, pp. 68–78, 1951.
[59] G. Tian, M. Zhou, J. Chu, T. Qiang, and H. S. Hu, "Stochastic cost-profit tradeoff model for locating an automotive service enterprise," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 2, pp. 580–587, Apr. 2015.
[60] G. Tian, M. Zhou, P. Li, C. Zhang, and H. Jia, "Multiobjective optimization models for locating vehicle inspection stations subject to stochastic demand, varying velocity and regional constraints," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 7, pp. 1978–1987, Jul. 2016.
[61] Y. Yin, T. C. E. Cheng, and D.-J. Wang, "Rescheduling on identical parallel machines with machine disruptions to minimize total completion time," *Eur. J. Oper. Res.*, vol. 252, pp. 737–749, Aug. 2016.
[62] B. Wang, X. Xia, H. Meng, and T. Li, "Bad-scenario-set robust optimization framework with two objectives for uncertain scheduling systems," *IEEE/CAA J. Autom. Sinica*, vol. 4, no. 1, pp. 143–153, Jan. 2017.
[63] F. Yang, K. Gao, I. W. Simon, Y. Zhu, and R. Su, "Decomposition methods for manufacturing system scheduling: A survey," *IEEE/CAA J. Autom. Sinica*, vol. 5, no. 2, pp. 389–400, Mar. 2018.



YAPING FU received the B.S. degree in commodity science from the Harbin University of Commerce, Harbin, China, in 2008, the M.S. degree in economics and management from Northeast Electric Power University, Jilin, China, in 2011, and the Ph.D. degree in systems engineering from Northeastern University, Shenyang, China, in 2015. He is currently an Associate Professor with the School of Business, Qingdao University, China. His research focuses on multi-objective production planning and scheduling, evolutionary multi-objective optimization, and simulation optimization. He has published over 20 journal and conference proceedings papers in the above research areas.



MENGCHU ZHOU (S'88–M'90–SM'93–F'03) received the B.S. degree in control engineering from the Nanjing University of Science and Technology, Nanjing, China, in 1983, the M.S. degree in automatic control from the Beijing Institute of Technology, Beijing, China, in 1986, and the Ph.D. degree in computer and systems engineering from the Rensselaer Polytechnic Institute, Troy, NY, USA, in 1990. He joined the New Jersey Institute of Technology (NJIT), Newark, NJ, USA, in 1990, where he is currently a Distinguished Professor of electrical and computer engineering. His research interests are in Petri nets, intelligent automation, the Internet of Things, big data, web services, and intelligent transportation. He has over 800 publications, including 12 books, over 460 journal papers (more than 360 in the IEEE transactions), 12 patents, and 28 book-chapters. He is a Life Member of the Chinese Association for Science and Technology-USA and served as its President, in 1999. He is a Fellow of the International Federation of Automatic Control (IFAC), the American Association for the Advancement of Science (AAAS), and the Chinese Association of Automation (CAA). He is a recipient of the Humboldt Research Award for the US Senior Scientists from the Alexander von Humboldt Foundation, the Franklin V. Taylor Memorial Award, and the Norbert Wiener Award from the IEEE Systems, Man and Cybernetics Society for which he serves as the VP for Conferences and Meetings. He is the founding Editor of the IEEE Press Book Series on Systems Science and Engineering, and the Editor-in-Chief of the IEEE/CAA Journal of Automatica Sinica.

XIWANG GUO (M'19) received the B.S. degree in computer science and technology from the Shenyang Institute of Engineering, Shenyang, China, in 2006, the M.S. degree in aeronautics and astronautics manufacturing engineering from Shenyang Aerospace University, Shenyang, in 2009, and the Ph.D. degree in system engineering from Northeastern University, Shenyang, in 2015. He was a Visiting Scholar with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA, in 2018. He is currently an Associate Professor with the College of Computer and Communication Engineering, Liaoning Shihua University. His research focuses on remanufacturing, recycling and reuse of automotive, and intelligent optimization algorithm. He has published over 30 journal and conference proceedings papers in the above research areas.

LIANG QI (S'16–M'18) received the B.S. degree in information and computing science and the M.S. degree in computer software and theory from the Shandong University of Science and Technology, Qingdao, China, in 2009 and 2012, respectively, and the Ph.D. degree in computer software and theory from Tongji University, Shanghai, China, in 2017. He is currently a Lecturer of computer science and technology with the Shandong University of Science and Technology. From 2015 to 2017, he was a Visiting Student with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA. He has authored over 20 technical papers in journals and conference proceedings, including the IEEE Transactions on System, Man and Cybernetics: Systems, the IEEE Transactions on Intelligent Transportation Systems, and the IEEE/CAA Journal of Automatica Sinica. His current research interests include Petri nets, discrete event systems, and optimization algorithms. He received the Best Student Paper Award-Finalist at the 15th IEEE International Conference on Networking, Sensing and Control (ICNSC'2018).