A Multi-Objective Cuckoo Search Algorithm Based on the Record Matrix for a Mixed-Model Assembly Line Car-Sequencing Problem

HUI WANG, BUYUN SHENG, QIBING LU, RUIPING LUO, AND GAOCAI FU
School of Mechanical and Electronic Engineer, Wuhan University of Technology, Wuhan 430070, China
Corresponding author: Qibing Lu (luqibing@whut.edu.cn)

This work was supported by the National Key Technology Research and Development Program of China under Grant 2016YFB1101700.

ABSTRACT In the car-sequencing problem of mixed-model assembly lines, a series of cars with different model types will be put into the assembly line in a certain order considering a variety of goals and constraints. In this paper, a multi-objective cuckoo search algorithm based on the record matrix is proposed to solve this problem. In this algorithm, the factors, including the variation of parts usage rates, variation of workstation workload, idle time, overload time, and model switching cost are considered. The record matrix proposed in this paper is utilized to record the characteristic information of the optimal solutions and historical solutions. Meanwhile, two search strategies based on the record matrix are proposed to enhance the ability of local search and global search in the algorithm. The proposed algorithm is verified by a real case. The results show that the proposed model and algorithm have good results, and they have the potential to address other similar problems.

INDEX TERMS Car-sequencing problem, mixed-model assembly line, multi-objective cuckoo search algorithm, record matrix.

I. INTRODUCTION Nowadays, it is an important challenge for automobile manufacturers to produce various cars with low costs. In the past, a single-model assembly line was adopted to produce a large quantity of cars, which can tremendously improve the efficiency of production. However, the single-model assembly line cannot satisfy the wide variety of customers demand. Meanwhile, competitive prices and the balance between man-power and machine lead to the production system developing toward the mixed-model assembly lines (MMAL). Increasing production systems are using MMAL because MMAL have more flexibility, better parts usage rates, and the ability to answer various demands of customers without possessing a large product inventory [1].

In a single-model assembly line, the model of each product on the line is the same. When products are put into a assembly line, there is no need to consider the sequencing problem. However, in the mixed-model assembly line, because of products differences in the structure configuration, shape, size, and assembly process parameters, the processing sequence of products will affect the operation efficiency of the assembly line. It is necessary to investigate how to arrange the sequence of various products for the mixed-model assembly line under the specific constraints and the planning goals. The sequence of products to be scheduled in the mixed-model assembly line is called the sequencing problem. According to different management strategies of enterprises, there are some universal optimization objective functions, including balancing the usage rate of parts, minimizing the idle time and overload time of the station, minimizing the total number of stops of the assembly line, and minimizing the model switching cost of the assembly line.

In the mixed-model assembly line, different kinds of products need different parts or different quantities of parts, which makes it difficult for a assembly line system to ensure the load balance of all production units and to keep the inventory at a low level. References [2], [3] adopted a maximum load ratio to prevent conflicts that many consecutive vehicles require the same option. Then, a constraint programming...
model was constructed. There is plenty of research focusing on the heuristic methods to address the car-sequencing model. Reference [4] developed a new color spread heuristic based on exact methods to solve the car sequencing problem. Reference [5] studied a set of heuristics for the car-sequencing problem, and four criteria were proposed to classify the heuristics, including branching, exploration, selection, and aggregation. The conclusion is that the most robust heuristics would be those branching on classes variables and selecting options. Reference [6] introduced regular expressions to describe the allowable patterns in the model of car sequencing problem. Then, an integer programming and a genetic algorithm were applied in this problem. In the real world, frequently perturbations occur, which stirs up an initially planned sequence. Therefore, Reference [7] investigated the re-sequencing problem, and a selectivity banks and a graph structure were utilized. The balance of production rate has an important influence on the operation efficiency of the mixed-model assembly line. Reference [8] utilized the smoothing variation of production rates as a substitute for the ultimate objective of smoothing parts usage rates under the assumption that products require approximately equal number and mix of parts. The uneven production rate is easy to cause the mixed-model assembly line shutting down, thus affecting the production efficiency of the assembly line. In the mixed-model assembly line, some workstations will complete the assembly within the circle time, thus the idle time is inevitably generated; while some workstations cannot complete the assembly within the circle time, thus the overload time is generated. In closed workstations, workers cannot enter the next workstation to continue to work, so another multi-skilled workers need to complete the rest assembly, thus the efficiency of assembly is decreased. Reference [9] analyzed the utility work caused by the overload of the workstation and optimized the sequencing problem of the mixed-model assembly line, considering minimizing the weighted sum of expected total work-overload and idleness. Reference [10] compared a mixed-model sequencing model with a car sequencing model for minimizing work overload, which was to quantify the gap in the solution quality between two models.

A majority of existing research on the mixed-model assembly line sequencing problem is concentrated on these factors, i.e. efficiency, cost, time, load, utility work, and system uncertainty [11]. In the mixed-model assembly line sequencing problem under the order oriented production environment, Reference [1] was concerned with six objectives, including the total utility work cost, total idle cost, total setup cost, total production rate variation cost, total operation error cost, and total tardiness cost for all orders. Then, particle swarm optimization (PSO) and simulated annealing (SA) algorithms were utilized to solve the problem. Reference [12] studied the sequencing problem of mixed-model assembly lines considering changeover complexity, and three changeover complexities evaluated with information entropy were proposed. Ant colony optimization (ACO) algorithm was adopted to minimize the number of violating sequencing rules and the level of changeover complexity. Reference [13] presented an integrated model to describe the asynchronous mixed-model assembly line combining balancing, sequencing, and buffer allocation. Reference [14] investigated the multi-criteria model sequencing problem, considering three factors: flow time, total duration, and idle time. An improved integrated smart multi-criterion Nawaz, Enscore, and Ham algorithm was presented to solve the problem. Reference [15] proposed a mixed-model sequencing problem of multi-station assembly line with stochastic processing time. The optimization objective was to minimize the weighted sum of expected total overload and idleness. A new hyper simulated annealing algorithm was developed to solve the problem, where the Q-learning algorithm was proposed to select the appropriate heuristic rules in the search process. Reference [16] studied a mixed-model assembly line sequencing problem focusing on the internal and external setup time. The target of the problem is to minimize the variation of production rate. The capacity and zero machine idle time constraints need to be satisfied. A modified intelligent bit mutation algorithm was developed to solve the sequencing problem. Reference [17] investigated a simultaneous optimization problem of order scheduling and mixed-model assembly line sequencing in assembly-to-order environment, considering three objectives: maximizing net profit earned from orders, reducing sequence-dependent setup time between different models, and leveling material usage. A multi-objective hybrid particle swarm optimization algorithm was proposed to solve the problem. Reference [18] studied a sequencing problem of hull mixed-model assembly line, optimizing two objectives: satisfaction ratio of delivery date and system complexity degree arising from its state frequent changes. An improved particle swarm optimization algorithm was developed to solve the problem. In addition, Reference [19] presented a sequencing problem of mixed-model assembly lines with station-dependent assembly times, concentrating on the minimization of the total utility work. A simulated annealing algorithm with two main hypotheses including simultaneous search and feasible search was presented to solve the problem.

There are various factors considered in the sequencing problem of mixed-model car assembly lines through the review of the research above and these factors have a common point, i.e. considering the level scheduling of assembly lines. Mixed-model car assembly lines produce cars with various models, and the car with different models need different operation duration in different workstation. The number of parts is also quite different for different models. Thus, an unreasonable production plan cause the assembly line assemble cars inefficient, even lead to the frequent shutdowns of the assembly line, which obtains the expensive cost. Moreover, the amount of the type of parts for producing one car is more than ten thousand, and it is important to design a reason-
able schedule for the mixed-model assembly line. Overall, considering level scheduling for mixed-model car assembly lines is necessary. Level scheduling is concerned with two main aspects: the level of parts and time. The level of parts is to address the level of the type and amount of the parts. Specifically, the number of the part types and amount of the parts need to be similar as more as possible. For the level of the time, the overload time and idle time of each workstation need to be short as more as possible. Car assembly lines can run efficiently with the pre-designed circle time only after considering the level scheduling.

The sequencing problem of mixed-model car assembly lines is NP-hard, thus, it is important to design an efficient algorithm. There are three kinds of algorithms to solve this problem, including exact algorithms, heuristic algorithms, and meta-heuristic algorithms. Exact algorithms are suitable for the problem with small scale, but it is time-consuming for the large-scale problem. Heuristic algorithms rely on the design of scheduling rules. However, it is not guaranteed that the optimal solution can be searched. Meta-heuristic algorithms can search the approximate optimal solutions through the iterative process and perform well, even in large-scale problems. Thus, increasing researchers focus on adopting meta-heuristic algorithms to solve this problem, which is a mainstream of the research method.

Mixed-model assembly line sequencing problems have been regarded as multi-objective optimization problems in the last decade. Many methods and models have been proposed, including multi-objective genetic algorithm [20], multi-objective simulated annealing algorithm [21], multi-objective ant colony optimization algorithm [22], and multi-objective artificial bee colony algorithm [23]. Reference [20] proposed a multi-objective genetic algorithm to solve the sequencing problem of fabrication production system, consisting of a mixed-model assembly line and a flexible manufacturing line. Two optimization objectives were considered simultaneously: minimizing the total variation of parts usage rates and makespan in the fabrication line. Reference [21] presented a simulated annealing meta-heuristic algorithm to address the mixed-model assembly line sequencing problem, and three objectives were optimized: idle cost, setup cost, and production rate variation cost. Reference [22] introduced a balancing and sequencing problem of mixed-model parallel two-sided assembly lines, then, an agent-based ant colony optimization was proposed to solve the problem. Reference [23] proposed a Pareto based multi-objective artificial bee colony algorithm for the simultaneous optimization of the balancing and sequencing problem of mixed-model assembly lines taking into account three objectives: balancing the workload of different models on each station, reducing the deviation of workload, and minimizing the total flow time of models on different stations.

Level scheduling plays an important role in assembly lines for efficiency and stability. Therefore, this paper makes a further investigation about the sequencing problem of mixed-model car assembly lines. In the aspect of the level scheduling, this paper considers the variation of parts usage rates, variation of station workload, idle time, overload time, and model switching cost to design the sequencing plan of mixed-model car assembly lines. A multi-objective optimization model of mixed-model assembly lines car-sequencing problem is constructed. To efficiently solve the sequencing scheduling of car assembly lines, a novel multi-objective optimization algorithm with a memory mechanism is proposed, i.e. the multi-objective cuckoo search algorithm based on the record matrix (RMMOCS). The record matrix is inspired by the phenomenon that most of meta-heuristic algorithms for solving multi-objective optimization models only focus on the objective value calculated by each solution, and they do not make full use of the historical information of generated solutions. In most multi-objective optimization algorithms, there are many optimal solutions in the iterative process, and how to select the optimal solution is essential for the search performance of the algorithm because it will become the evolution direction of the next generation solution. The record matrix in the algorithm, including the optimal record matrix and history record matrix, can record the information of multiple optimal solutions and new generated solutions. The cuckoo search algorithm is a meta-heuristic algorithm which simulates the breeding behavior of some cuckoos and the flight behavior of some birds or fruit flies. The research shows that its search performance is better than some existing algorithms such as genetic algorithm and particle swarm optimization algorithm [24]. Based on the record matrix, this paper proposes two search strategies: the Levy flight strategy based on the optimal record matrix and empty nest search strategy based on the history matrix. The optimal record matrix can guide the algorithm referencing the information of multiple optimal solutions, and then the local search ability of the algorithm is strengthened. The history matrix can help the algorithm explore the unknown solution space, which enhances the global search ability of the algorithm.

The remainder of the paper is organized as follows. In Section II, a sequencing problem of mixed-model assembly lines and a mathematical model are described. The cuckoo search algorithm based on the record matrix is introduced in Section III. Experimental result and comparison are shown in Section IV. Finally, conclusion remarks and future works are provided in Section V.

II. MATHEMATICAL MODEL
A. PROBLEM DESCRIPTION

The assembly workshop is the last step in the production process of passenger cars. The assembly workshop adopts mixed-model assembly lines to assemble various types of passenger car bodies. The painted passenger car bodies are transported to each station of the assembly workshop through the conveyor line, and the parts required for the assembly of passenger cars are delivered to the assembly line according to the plan information. In the assembly main lines, the passenger car bodies are consecutively assembled in the interior
In the mixed-model car assembly line of the assembly workshop, there are \( I \) car bodies that will be assembled at \( K \) workstations in turn. There are \( J \) models of car body, and the number of each model is \( n_j \). Let \( d_{ijk} \) be the assembly duration of the \( i \)th car body with model \( j \) assembled at workstation \( k \). \( x_{ij} \) is the decision variable, if the model of the \( i \)th car body is \( j \), \( x_{ij} = 1 \); otherwise, \( x_{ij} = 0 \). For the mixed-model car assembly line of the assembly workshop, this paper comprehensively considers the variation of parts usage rates, variation of workstation workload, idle time, overload time, and model switching cost to solve the car-sequencing problem.

The model built in this paper considers the following assumptions. There is no machine breakdown, and materials are enough. There are multi-skilled workers and machines. The position of the car body is fixed in the transportation system. There is no buffer between two adjacent workstations. An idle operator cannot be employed at another workstation. The traveling time of workers is not considered.

### B. THE VARIATION OF PARTS USAGE RATES

In the process of car body assembly, the parts to be assembled are transported to the workers according to the assembly production plan. Cars with different models or different configuration have different demands for the same parts on the same station, as shown in Fig. 2. For example, in the interior line, cars with different models have different requirements for the number of audio equipment and number of sunroof. If a certain type of car body with a large demand for audio equipment is assembled consecutively with a large number, the logistics demand for audio equipment will surge. Similarly, in a period of time, if there is no sunroof demand for some consecutive cars, the logistics demand for the sunroof will decrease. The variation of parts usage rates will not only bring variation to the logistics demand, but also affect the balance of parts in the line side warehouse, which will affect the operation stability of the production line. Moreover, the production line cannot realize the level production. To ensure the balance of parts usage on the line, it is necessary to optimize the car body sequence and to reduce the variation of parts usage rates as much as possible. The sum of the maximum variation of each part usage rate, \( M \), can be calculated by equations (1) and (2), where \( E \) is the total number of part type, \( R_{jke} \) represents the quantity of part type \( e \) consumed by model \( j \) on workstation \( k \), and \( Q_e \) denotes the ideal usage rate of part type \( e \) in a production sequence. In equation (1), the first item is the ideal usage for part type \( e \) during top \( z \) stages, and the second item is the actual usage for part type \( e \) during top \( z \) stages.

\[
M = \sum_{e=1}^{E} \max_{i=1}^{z} \sum_{k=1}^{K} \left| \sum_{j=1}^{J} R_{jke} \cdot x_{ij} - \sum_{j=1}^{J} Q_e \right| \quad (1)
\]

\[
Q_e = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} R_{jke} \cdot x_{ij} / I \quad (2)
\]

### C. THE VARIATION OF WORKSTATION WORKLOAD

In the actual assembly process, operators may need to simply adjust some tools and system parameters of assembly line when switching different models. For instance, the assembly workstation needs to make an adjustment when the type of car door changes. In group assembly, if the tooling and system parameters adopted by the parts of the same group are similar, the adjustment time is short when switching models; while the adjustment time of parts of different groups is relatively long. The variation of the actual load of the workstation will affect the operation smoothness of the assembly line, and the larger variation will greatly reduce the operation stability of the assembly line. Therefore, it is necessary to organize the sequence of car bodies reasonably to reduce the variation of the workstation workload.

Let \( t_{jk} \) be the adjustment time for switching model \( j \) at workstation \( k \). Due to the need to adjust the machine parameters, fixtures, and operation modes when assembling the car bodies of different models, extra adjustment time is added. The actual processing time of each workstation is computed by equation (3), where \( y(d_{j1}^{-1}, d_{j}^1) \) indicates whether models of adjacent cars are the same, if no, \( y(d_{j1}^{-1}, d_{j}^1) = 1 \); otherwise, \( y(d_{j1}^{-1}, d_{j}^1) = 0 \). Then, the variation of the workload of all workstations for a production sequence, \( W \), can be formulated by equation (4).

\[
d_{ijk} = d_{ijk} + t_{jk} \cdot y(d_{j1}^{-1}, d_{j}^1) \quad (3)
\]
D. IDLE TIME AND OVERLOAD TIME

When several kinds of car bodies are put into the mixed-model assembly line, some workstations workload may be greater than the circle time, while some may be less than the circle time. Therefore, different sequences of car bodies will influence the balance of workstations workload. There will be a certain amount of idle time in some workstations during the assembly process, and too much idle time will inevitably reduce the production efficiency of the assembly line. Meanwhile, some workstations will be overloaded. For excessive overload time, the assembly lines often need to be stopped. The shutdown caused by the overload workstation will also affect the efficiency of the assembly line. In addition, too frequent overload time will also make a bad influence on product quality. Therefore, reducing the idle time and overload time can improve the assembly efficiency and product quality.

The start time for the first car body to be assembled on the first workstation of the assembly line is $ES_{1jk} = 0$. When the first car is to be assembled on the $k$th workstation of the assembly line, its start time need to consider whether the front car body is overloaded on the subsequent workstations. The calculation of the start time $ES_{ijk}$ and finish time $EF_{ijk}$ is given in equations (5) and (6), where $C_a$ is the circle time of the assembly line.

$$ES_{ijk} = \max((k-1) \cdot C_a, EF_{ij(k-1)})$$  
$$EF_{ijk} = ES_{ijk} + d_{rijk}$$

When the $i$th car body is to be assembled on the subsequent workstation $k$, it is necessary to consider the overload of other car bodies and other workstations at the same time, i.e., judge whether or not the $i$th car body is overloaded at the $k-1$ th workstation and whether or not the $i-1$ th car body is overloaded at the $k$th workstation, then, the calculation of $ES_{ijk}$ and $EF_{ijk}$ is shown in equations (7) and (8).

$$ES_{ijk} = \max((i + k - 2) \cdot C_a, EF_{ij(k-1)}, EF_{(i-1)jk})$$  
$$EF_{ijk} = ES_{ijk} + d_{rijk}$$

The sum of idle time of the workstation, $T_i$, is calculated in equation (9). When a certain workstation is overloaded, the finish time of subsequent cars will be affected, i.e., the finish time of subsequent cars will be delayed to some extent. The calculation of overload time, $T_o$, is shown in equation (10), and the sum of idle time and overload time, $T$, is shown in equation (11).

$$T_i = \sum_{i=1}^{K} \sum_{k=1}^{K} \max(0, (i + k - 1) \cdot C_a - EF_{ijk})$$  
$$T_o = \sum_{i=1}^{K} \sum_{k=1}^{K} \max(0, EF_{ijk} - (i + k - 1) \cdot C_a)$$  
$$T = T_i + T_o$$

E. MODEL SWITCHING COST

In the process of switching different models, it is inevitable to adjust some machine parameters, fixtures, and operation methods. Apparently, the adjustment process will consume cost. For instance, different car models have different tire sizes, so it is necessary to adjust the grasping distance of the tire manipulator to adapt to the corresponding model tires, and the adjustment cost will increase in the adjustment process. Furthermore, when assembling different models, if their corresponding parts parameters are similar, the adjustment of fixtures or machine parameters is relatively small; otherwise, the adjustment will be large. Therefore, the reasonable organization of production sequence can effectively reduce the model switching cost.

Equation (12) is to compute the model switching cost $P$, where $w_{jk}$ is the cost unit of switching model $j$ at workstation $k$.

$$P = \sum_{k=1}^{K} \sum_{i=2}^{K} \gamma(d_j^{-1}, d_j^i) \cdot w_{jk}$$

F. OPTIMIZATION OBJECTIVE

In this paper, two optimization objectives are considered, including minimizing the total cost and time, as shown in equations (13) and (14). Equation (13) shows that the variation of parts usage rates, variation of workstation workload, and car model switching will be converted into the objective.
of minimizing the cost through the penalty cost unit and weighted coefficients, where \( \varepsilon \) and \( \tau \) are penalty cost unit of the variation of parts usage rates \( M \) and workstation workload \( W \) respectively; \( \alpha, \phi, \) and \( \gamma \) are the weighted cost coefficients. Equation (14) denotes minimizing the sum of idle time and overload time.

\[
\begin{align*}
    f_1 &= \text{Min}(\alpha \cdot \varepsilon \cdot M + \phi \cdot \tau \cdot W + \gamma \cdot P) \quad (13) \\
    f_2 &= \text{Min}(T) \quad (14)
\end{align*}
\]

**G. CONSTRAINT**

The proposed model considers the following constraints:

\[
\begin{align*}
    n_j &= \sum_{i=1}^{I} x_{ij}, \forall j = 1, 2, \ldots, J \quad (15) \\
    I &= \sum_{j=1}^{J} n_j \quad (16) \\
    ES_{ijk} &\geq EF_{(i-1)jk}, \quad \forall i = 1, 2, \ldots, I; \quad j_1, j_2 = 1, 2, \ldots, J; \quad k = 1, 2, \ldots, K \quad (17) \\
    ES_{ijk} &\geq EF_{ij(k-1)}, \quad \forall i = 1, 2, \ldots, I; \quad j = 1, 2, \ldots, J; \quad k = 1, 2, \ldots, K \quad (18) \\
    \sum_{i=z}^{i=z+I_{max}} x_{ij} &\leq I_{max}, \quad \forall z = 1, 2, \ldots, I - I_{max}; \quad j = 1, 2, \ldots, J \quad (19)
\end{align*}
\]

Equation (15) represents the total quantity constraint for each car model. Equation (16) shows that the sum of the number of each car model is equal to the total car body number. Equation (17) indicates that the next car body can be assembled on the workstation only after the previous car body is assembled. Equation (18) denotes that the car body cannot be assembled on the next workstation until the assembly process of the previous workstation is completed. Equation (19) is the maximum consecutive quantity constraint for the same car model, where \( I_{max} \) is the maximum consecutive number for each model in the sequence.

**III. METHODOLOGY**

**A. THE FRAMEWORK OF RMMOCS ALGORITHM**

In this paper, a multi-objective cuckoo search algorithm based on the record matrix is proposed. The algorithm consists of four main steps: Levy flight, empty nest search, history information update, and environment selection. This algorithm possesses the characteristics of memorizing and learning the historical search process. By constructing the optimal record matrix and the history record matrix, the solutions generated during the algorithm iteration are recorded. The record method is to map the generated solutions to specific matrices, and sum them with history record matrix or optimal record matrix. Levy flight is a search mechanism which has the characteristic of local search, i.e., each solution will evolve towards the optimal solution in history. However, for multi-objective algorithms, there exist numerous optimal solutions generated in the algorithm iteration. Choosing the appropriate optimal solution plays an important role in the subsequent algorithm optimization. In this paper, the selection strategy based on the optimal record matrix is proposed, and the information of multiple optimal solutions is referenced to obtain the best evolutionary direction. Empty nest search is a search process with the feature of global search. It aims to explore the solution space that has not been searched and it prevents the algorithm from falling into local convergence. Global search is usually realized by adopting random strategy. However, the random search strategy is not always efficient. This paper proposes a selection strategy based on the history record matrix to substitute for the random search strategy, which has a good exploration ability making the search process towards the unknown solution space. The history record update means to update the new information of solutions to optimal record matrix or history record matrix. Environment selection is a mechanism that evaluates the quality of the parent solutions and offspring solutions. This paper utilizes the method of non-dominated solution ranking and crowded degree to measure the quality of solutions.

Fig. 3 is the flowchart of the RMMOCS algorithm. The specific steps of the proposed methods are as follows:

**Step 1**: Initialize algorithm parameters, optimal record matrix, and history record matrix.

**Step 2**: Perform the Levy flight operation to generate new solutions, compare the parent solutions and offspring solutions, and select better solutions.

**Step 3**: Perform the empty nest search operation to generate new solutions, compare the parent solutions and offspring solutions, and select better solutions.
Step 4: Record the non-dominated solutions, and update the optimal record matrix and the history record matrix.

Step 5: Execute the environment selection operation to obtain the elite solutions.

Step 6: Judge whether the maximum iteration is reached. If yes, output Pareto solution set; otherwise, go to step 2.

B. CHROMOSOME CODING

In this paper, the priority index is utilized to describe the production sequence of car bodies with different models. The car body with a higher priority (lower value) is assembled first. Fig. 4 is an example of the chromosome coding. Assume that there are three models (model 1, 2, and 3) of car body to be assembled, and the number of each model is two. The scheduling part represents the priority permutation which corresponds to the models of car body. It can be seen from Fig. 4 that the original car body sequence (112233) to be produced is adjusted (212313) according to the priority (253164), i.e., the first assembled car body is model 2, the second model 1, the third model 2, the fourth model 3, the fifth model 1, and the last is model 3.

C. RECORD MATRIX

1) OPTIMAL RECORD MATRIX

The purpose of the optimal record matrix is to make the algorithm move in a better direction. For the multi-objective optimization model, there are numerous optimal solutions, but it is impossible to find the best optimal solution, which makes it difficult for some algorithms that rely on the optimal solution as the reference direction to search. In this paper, the optimal record matrix is proposed, and the information of several optimal solutions sequence is referenced to provide high-quality gene sequence for the subsequent search.

The optimal record matrix $G$ is an $n$-order matrix, where $n$ equals the length of a chromosome. Each element in the optimal record matrix $G$ corresponds to a gene in a chromosome. The index of columns in the matrix corresponds to the genes position in the chromosome, and the index of rows in the matrix denotes the specific gene codes. Each optimal solution needs to be converted into a matrix, and then it can be recorded in the optimal record matrix. Given a chromosome, it can always be mapped into a matrix according to its gene code and gene location. Assume there is a chromosome coding sequence $s$, it can be mapped into a matrix $A = \{a_{ij}\}$, and the mapping rules are given in equations (20), (21), and (22), where $f_0(s)$ represents the $0$th objective value of the chromosome $s$. The diagram of a chromosome mapped to a matrix is shown in Fig. 5. For example, the first, third, and sixth gene of the chromosome encoding (132465) in Fig. 5 are 1, 2, and 5, which denotes the elements $a_{11}, a_{23}, a_{56}$ of the matrix $A$ are selected. The selected element in the matrix equals the sum of multi objective values of the corresponding solution, and the unselected gene is represented by a value of zero in the matrix. Here, the sum of multiple objective values is reflection of the quality level of genes. The smaller the value is, the better the genes are.

\[
\begin{align*}
&v = \sum_{\alpha=0}^{\alpha=O} f_\alpha(s) \quad (20) \\
&a_{ij} = \begin{cases} 
&v, \text{ if } s[j] = i \\
&0, \text{ otherwise } \end{cases} \quad (21) \\
&\sum_{i=1}^{i=n} a_{ij} = v, \forall j; \quad \sum_{j=1}^{j=n} a_{ij} = v, \forall i \quad (22)
\end{align*}
\]

After each iteration, the new solutions will be sorted by the non-dominated solutions ranking (see Section 3.6) to select the better solutions, which will then be mapped to the matrix. After the non-dominated solutions are converted into matrices, they are added to the optimal record matrix $G$ by equations (23) and (24), and then updating the optimal record matrix is completed. In equations (23) and (24), $l$ is the current number of iterations, $z$ is the index of solutions, and Rou is the forgetting coefficient. By introducing forgetting coefficient, it can effectively prevent the algorithm from falling into the dilemma of local optimization [22].

The diagram of updating the optimal record matrix is shown in Fig. 6. For example, five optimal chromosomes (142356, 415623, 253164, and 564231, 253164, and 415623) in Fig. 6 are first mapped to five matrices $A_1, A_2, A_3, A_4, A_5$, and then the sum of five matrices will be added to the matrix $G$.

\[
\begin{align*}
&g^l_{ij} = g^{l-1}_{ij} \cdot (1 - Rou) + \sum_{z=1}^{z=Z} a^l_{ij} \quad (23) \\
&G^l = G^{l-1} \cdot (1 - Rou) + \sum_{z=1}^{z=Z} ConG(s_z) \quad (24)
\end{align*}
\]
the algorithm. The history record matrix not only effectively prevents the algorithm from falling into local spaces as much as possible in the search process, which not presented to make the algorithm explore more unknown solutionoptimal record matrix. The history record matrix process of iteration, and its record way is different from the

2) HISTORY RECORD MATRIX

The history record matrix is similar to the optimal record matrix, but it records all solutions generated during the process of iteration, and its record way is different from the optimal record matrix. The history record matrix \( H \) is presented to make the algorithm explore more unknown solution spaces as much as possible in the search process, which not only effectively prevents the algorithm from falling into local convergence, but also improves the global search ability of the algorithm. The history record matrix \( H \) is also an \( n \)-order matrix. Assume there is a chromosome coding sequence \( s \), it can be mapped into a matrix \( B = \{b_{ij}\} = ConH(s) \), as shown in Fig. 7. Different from the mapping rules in Section III-C-1, the mapped element values in the matrix equal one, and the mapping rules are shown in equations (25) and (26). Here, the element of the matrix represents the frequency of gene occurrence.

\[
b_{ij} = \begin{cases} 
1, & \text{if } s[j] = i \\
0, & \text{otherwise}
\end{cases} \tag{25}
\]

\[
\sum_{i=1}^{n} b_{ij} = 1, \quad \forall j; \quad \sum_{j=1}^{n} b_{ij} = 1, \quad \forall i \tag{26}
\]

In the iterative process, the algorithm will generate solutions continually, including the repeated solutions, but these repeated solutions cannot bring further optimization to the search. Therefore, this paper proposes the history record matrix to solve this problem. The specific implementation method is to map the solution generated in the iteration process into a matrix and add the matrix to the history record matrix \( H \), as shown in equations (27) and (28), then updating the history record matrix is completed, as shown in Fig. 8. For example, five chromosomes \((324615, 136254, 561342, 652431, \text{and} 245163)\) in Fig. 8 are first mapped to five matrices \((B1, B2, B3, B4, \text{and} B5)\), and then the sum of five matrices will be added to the matrix \( H \). It can be seen from Fig. 8 that the generated solutions will be recorded in the history record matrix, where the corresponding element in the history record matrix plus 1 is calculated. In the subsequent search of the algorithm, unexplored genes with a lower value in the history record matrix will be generated preferentially, which is described in Section III-E.

\[
h_{ij} = \sum_{z=1}^{Z} b_{ij}(s_z) \tag{27}
\]

\[
H = \sum_{z=1}^{Z} ConH(s_z) \tag{28}
\]

D. LEVY FLIGHT STRATEGY BASED ON OPTIMAL RECORD MATRIX

Levy flight operation is a kind of flight strategy simulating some birds and fruit flies. The consecutive steps generated by the Levy flight operation essentially form a random walk process which obeys a power-law step-length distribution with a heavy tail [25]. In the intelligent optimization algorithm, Levy flight can expand the search range and increase the population diversity. The Levy flight operation aims at iterative evolution towards a better direction. The Levy flight strategy based on the optimal record matrix adopts the Levy flight operation to generate the corresponding step length in each dimension, as shown in equations (29), (30), (31), (32), and (33); then, the step length generated by each dimension is utilized to select the gene set from the optimal record matrix, where the number of gene in the gene set is equal to the step length of each dimension; finally, the roulette selection strategy is applied for the selected gene set to obtain the high-quality gene sequence. The specific steps are described as follows.

**Step 1**: According to the optimal record matrix \( G \), obtain the gene set of each column \( Gene[j] = \{g[j, 1], g[j, 2], \ldots, g[j, n]\} (j = 1, 2, \ldots, n) \).

**Step 2**: Sort each set \( Gene[j] (j = 1, 2, \ldots, n) \) in a descending order, and set the result to \( Gene[j] (j = 1, 2, \ldots, n) \).

**Step 3**: Calculate the step length \( (step(j)) \) of each dimension of a solution by equations (29), (30), (31), (32), and (33); then, select the top \( step(j) \) elements in \( Gene[j] \), denoted as \( BestGene[j] (j = 1, 2, \ldots, step(j)) \), and record the corresponding gene code \( GeneCode[j] (j = 1, 2, \ldots, step(j)) \).
Step 4: Calculate cumulative probability \( \text{Probability}[j] \) according to the set \( \text{BestGene}[j] (j = 1, 2, \ldots, step(j)) \), perform roulette operation in turn, and obtain the gene sequence without repetition.

\[
\begin{align*}
\text{step}(j) & = \left( \frac{u(j)}{v(j)} \right)^{1/\beta} \quad (29) \\
u(j) & = \text{rand}(0, 1) \cdot J \quad (30) \\
v(j) & = \text{rand}(0, 1) \cdot J \quad (31) \\
\sigma & = \left( \frac{\Gamma(1 + \beta) \cdot \sin((\pi \cdot \beta)/2)}{\Gamma((1 + \beta)/2) \cdot \beta \cdot 2^{(\beta-1)/2}} \right)^{1/\beta} \quad (32) \\
\text{step}(j) & = \begin{cases} 
J, & \text{if Ceiling}(\text{step}(j)) > J \\
\text{Ceiling}(\text{step}(j)), & \text{otherwise} 
\end{cases} \quad (33)
\end{align*}
\]

where \( \beta = 3/2, \Gamma(a) = \int_0^{+\infty} e^{-t} \cdot t^{a-1} \, dt, \text{rand}(0, 1) \) means to randomly generate a decimal between 0 and 1, and Ceiling(·) denotes the minimum integer greater than or equal the specified value.

After obtaining the best gene sequence, the parent solution needs to fuse with the obtained gene sequence, as shown in Fig. 9. First, two points as the start and end positions of the gene fusion are randomly selected. Then, the parent solution is replaced with the best gene sequence obtained above. Finally, duplicate genes are resolved by randomly generating non-duplicate genes. For example, in Fig. 9, the start and end positions are selected as 3 and 5, thus the gene sequence (451) of the best gene. It can be seen that there are two repeated genes 5 in the new solution. Therefore, one of the repeated genes 5 will be adjusted.

After the new solution is generated, it is necessary to evaluate the quality of the parent and offspring solutions, mainly utilizing the dominant relationship to evaluate. The specific steps are as follows:

Step 1: Judge whether the offspring solution dominates the parent solution. If yes, the parent solution will be replaced with the offspring solution, and the evaluation is finished; otherwise, go to step 2.

Step 2: Judge whether \( \text{rand}(0, 1) < 0.5 \). If yes, the parent solution will be replaced with the offspring solution; otherwise, the parent solution will not be replaced.

E. EMPTY NEST SEARCH STRATEGY BASED ON HISTORY RECORD MATRIX

Empty nest search aims to explore new solution spaces. In this paper, the information of the solution recorded by the history matrix is utilized to guide the algorithm towards unexplored solution spaces. Specifically, the empty nest search strategy based on the history record matrix utilizes the frequency information of each gene stored in the history record matrix, guiding the algorithm to produce gene sequences with lower frequency as much as possible. Since the genes with lower frequency are selected preferentially, it is necessary to process the elements in the history matrix by equation (34). After generating the step length of each dimension by equation (35), the gene set is selected from the history matrix, where the number of gene in gene set is equal to the step length, and finally the roulette strategy to obtain the gene of each dimension is executed. The specific steps are as follows:

Step 1: Obtain the gene set of each column \( \text{Gene}[j] = \{h_{ij}, h_{ij}, \ldots, h_{ij}\} (j = 1, 2, \ldots, n) \) according to the history record matrix \( H \).

Step 2: Address \( \text{Gene}[j] (j = 1, 2, \ldots, n) \) by equation (34).

Step 3: Sort each set \( \text{Gene}[j] (j = 1, 2, \ldots, n) \) in a descending order, and set the result to \( \text{Gene}[j] (j = 1, 2, \ldots, n) \).

Step 4: Calculate the step length (\( \text{step}(j) \)) of each dimension of a solution by equation (35), select the top \( \text{step}(j) \) elements in \( \text{Gene}[j] \), denoted as \( \text{ExploreGene}[j] (j = 1, 2, \ldots, \text{step}(j)) \), and record the corresponding \( \text{GeneCode}[j] (j = 1, 2, \ldots, \text{step}(j)) \).

Step 5: According to \( \text{ExploreGene}[j] \), calculate the cumulative probability \( \text{Probability}[j] \), perform roulette operation in turn, and obtain the gene sequence without repetition.

\[
\begin{align*}
h_{ij} & = 1/(h_{ij} + 1) \quad (34) \\
\text{step}(j) & = \text{rand}(0, 1) \cdot J \quad (35)
\end{align*}
\]

After the exploration gene sequence is obtained, the parent solution needs to exchange gene with the exploration gene sequence, as shown in Fig. 10. First, the position of exchanging gene is determined by equation (36), where a value of one means the corresponding gene will be exchanged; otherwise, no exchange occurs. Then, the newly generated gene sequence is replaced into the parent solution. Finally, duplicate genes are resolved by randomly generating the genes without repetition. For example, in Fig. 10, the select sequence (011001) determines the position of gene to be
exchanged (2, 3, and 6). Therefore, the genes (3, 4, and 6) of the parent solution will be replaced with the genes (6, 2, and 3) of the explore gene. It can be found that there are two repeated genes 2 in the new solution. Thus, one of repeated genes 2 will be adjusted.

After the new solution is generated, the evaluating operation between the parent solution and the offspring solution will be executed. The evaluation strategy is the same as that described in Section III-D.

$$select(j) = \begin{cases} 1, & \text{if } \text{rand}(0, 1) < P_a \\ 0, & \text{otherwise} \end{cases} \quad (36)$$

where $P_a$ is the probability of finding a new nest.

**F. ENVIRONMENT SELECTION**

Environment selection is to evaluate the solutions of the current generation and the elite solutions of the previous generation, and the better solutions will be selected for the next iteration. The evaluation strategy adopted in this paper is non-dominated solution ranking and crowded degree computing [26].

In NSGA-II, the non-dominant level represents the dominant relationship of individuals in the population, and the larger level (smaller value) denotes the solution is better. The specific steps of calculating the non-dominant level are as follows:

**Step 1:** Initialize $\text{rank} = 1$.

**Step 2:** Traverse each solution in the population $U$ to determine whether the solution $s_n$ ($s_n$ belongs to $U$) is not dominated by any solution. If yes, set $\text{rank}$ as the non-dominated level of the solution $s_n$.

**Step 3:** Remove the solutions from $U$ whose non-dominance level are determined.

**Step 4:** Judge whether there is any non-dominated level of solution that is not determined in $U$. If yes, update $\text{rank} = \text{rank} + 1$, and go to step 2; otherwise, terminate.
The crowded degree is utilized to evaluate the quality of the solutions with the same dominant level. A larger evaluation value indicates a better distribution of the solution. The crowded degree of the solution is calculated as follows:

**Step 1**: Determine the non-dominant level of population $U$, and initialize $\text{rank} = 1$.

**Step 2**: Obtain the individuals with $\text{rank}$ level, and sort them according to the objective value.

**TABLE 3.** Parts required for different models.

| Model | Part 1 | Part 2 | Part 3 | Part 4 | Part 5 | Part 6 | Part 7 | Part 8 | Part 9 | Part 10 |
|-------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| A     | 2      | 1      | 3      | 4      | 0      | 2      | 0      | 1      | 5      | 4      |
| B     | 2      | 2      | 0      | 5      | 2      | 0      | 3      | 2      | 2      | 5      |
| C     | 2      | 1      | 0      | 3      | 2      | 0      | 2      | 1      | 4      | 3      |
| D     | 2      | 0      | 2      | 5      | 0      | 1      | 2      | 3      | 0      | 4      |
| E     | 0      | 3      | 4      | 0      | 4      | 3      | 0      | 0      | 5      | 0      |
| F     | 3      | 0      | 2      | 4      | 0      | 4      | 3      | 5      | 0      | 2      |
| G     | 3      | 3      | 5      | 5      | 0      | 2      | 0      | 4      | 6      | 0      |
| H     | 0      | 4      | 2      | 3      | 6      | 3      | 4      | 0      | 2      | 3      |
| I     | 4      | 5      | 3      | 0      | 5      | 0      | 3      | 2      | 3      | 4      |
| J     | 5      | 3      | 2      | 1      | 0      | 4      | 5      | 3      | 4      | 0      |

**TABLE 4.** Adjustment cost of different models at different stations.

| Model | Station 1 | Station 2 | Station 3 | Station 4 | Station 5 | Station 6 | Station 7 | Station 8 | Station 9 | Station 10 |
|-------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|------------|
| A     | 1         | 3         | 5         | 4         | 1         | 2         | 1         | 6         | 1         | 7          |
| B     | 2         | 1         | 3         | 5         | 2         | 1         | 6         | 1         | 7         | 1          |
| C     | 2         | 4         | 1         | 6         | 4         | 7         | 2         | 4         | 6         | 1          |
| D     | 1         | 4         | 6         | 2         | 1         | 6         | 1         | 7         | 1         | 9          |
| E     | 3         | 2         | 4         | 1         | 3         | 5         | 3         | 2         | 4         | 3          |
| F     | 4         | 3         | 2         | 3         | 3         | 4         | 4         | 3         | 2         | 2          |
| G     | 2         | 5         | 5         | 4         | 3         | 4         | 5         | 2         | 1         | 5          |
| H     | 3         | 4         | 6         | 3         | 2         | 6         | 4         | 4         | 3         | 2          |
| I     | 2         | 3         | 2         | 2         | 3         | 5         | 3         | 2         | 1         | 3          |
| J     | 4         | 3         | 1         | 2         | 5         | 6         | 7         | 4         | 5         | 3          |

**TABLE 5.** Production quantity of different models.

| Instance | Model A | Model B | Model C | Model D | Model E | Model F | Model G | Model H | Model I | Model J |
|----------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| 20x4     | 4       | 6       | 5       | 5       | 0       | 0       | 0       | 0       | 0       | 0       |
| 30x6     | 5       | 7       | 4       | 6       | 4       | 4       | 0       | 0       | 0       | 0       |
| 40x8     | 6       | 5       | 6       | 4       | 5       | 6       | 4       | 4       | 0       | 0       |
| 50x10    | 4       | 5       | 6       | 5       | 6       | 4       | 5       | 6       | 5       | 4       |
Step 3: Calculate crowded degree for each solution with rank level according to equation (37) [26].

Step 4: Judge whether there are other solutions in $U$ that do not determine the crowded degree. If yes, update $\text{rank} = \text{rank} + 1$ and go to step 2; otherwise, terminate.

$$Cd_n = \begin{cases} +\infty, & \text{if } n = 1 \text{ or } n = N \\ \sum_{o=1}^{O} f_{o}(s_{n+1}) - f_{o}(s_{n-1}) \quad \text{otherwise} \end{cases}$$ (37)

where $o$ represents the index of objectives, $f_{o}(s_{n})$ denotes the $o$th objective value of solution $s_{n}$, $f_{o\text{max}}$ and $f_{o\text{min}}$ represent the maximum and minimum of the $o$th objective value, respectively.

IV. NUMERICAL EXPERIMENT

In this section, the effectiveness and performance of RMMOCS algorithm are illustrated by numerical experiments. First, the proposed algorithm will be compared with multi-objective cuckoo search (MOCS) algorithm and NSGA-II algorithm. RMMOCS and MOCS algorithms have the same algorithm framework, and the environment selection mechanism of RMMOCS is the same as NSGA-II, so the comparison experiment is conducive to verify the effectiveness of the search strategy based on the record matrix. Then, to further illustrate the performance of proposed algorithm, RMMOCS algorithm will be further compared with the state-of-the-art multi-objective optimization algorithms, including multi-objective artificial bee colony algorithm (MOABC) [23], multi-objective firefly algorithm (MOFA) [27], and multi-objective ant colony optimization algorithm (MOACO) [22]. These algorithms are written in C# language. The experiment will be carried out in Microsoft Visual Studio 2012 on Windows 7 system with CPU frequency of 2.4GHz and memory of 8G.

A. CASE CONSTRUCTION

This paper investigates an automobile manufacturer in Wuhan as a case study. The case enterprise mainly produces passenger cars, and the assembly workshop of the enterprise assembles various types of passenger cars by utilizing mixed-model assembly lines. In this paper, the typical mixed-model assembly process in the assembly workshop of this case is employed to conduct the scheduling experiments. The assembly duration of different models at different workstations are shown in Table 1. The adjustment time of switching different models at different workstations are presented in Table 2. The number of parts required for the assembly of different models is shown in Table 3. The adjustment cost of switching different models at different workstations is shown in Table 4. Table 5 shows the cars models and quantities to be produced, which are utilized as scheduling instances. The circle time $C_a$ is set to 112. The maximum number of consecutive quantity for each model $I_{\text{max}}$ is set to 3. The penalty cost unit of the

![FIGURE 11](image-url). The best Pareto front of different algorithms. (a) Instance 4 x 20. (b) Instance 6 x 30. (c) Instance 8 x 40. (d) Instance 10 x 50.
variation of the parts usage rates $\varepsilon$ and workstation workload $\tau$ are set to 3.5 and 2.5 respectively. The weighted coefficients $\alpha, \phi$ and $\gamma$ are set to 1/3.

**B. PERFORMANCE EVALUATION METRICS**

The performance evaluation of multi-objective algorithms is mainly to evaluate the quality of Pareto solutions. In this paper, three universal multi-objective performance evaluation metrics are selected to evaluate the algorithms, i.e., general distance (GD), spread distance (SP), and hyper volume (HV). GD is utilized to evaluate the convergence of the solutions, SP is adopted to evaluate the distribution of the solutions, and HV is employed to simultaneously evaluate the convergence and distribution of the solutions.

(1) General distance (GD): This metric is utilized to evaluate the average minimum distance between the solutions obtained by the algorithm and the ideal solutions. The lower the evaluation value, the better the performance of the algorithm. The formulation of GD is shown in equation (38) [28], [29], where $N$ is the number of non-dominated solutions $s_n$, $PF^*$ includes the ideal points of all optimization objectives, and $\text{dis}(s_n, s^*)$ represents the Euclidean distance between solution $s_n$ and $s^*$.

$$GD = \frac{1}{N} \sum_{n=1}^{N} \min_{s^* \in PF^*} (\text{dis}(s_n, s^*))^2$$ (38)

(2) Spread distance (SP): This metric evaluates the standard deviation of the minimum distance from each solution to other solutions. The lower the evaluation value indicates that solutions have better distribution performance. The calculation of SP is shown in equations (39) and (40) [29], [30], where $d_n$ is the minimum distance from the $n$th solution to other solutions, and $\bar{d}$ represents the mean value of $d_n$.

$$SP = \sqrt{\frac{1}{N-1} \sum_{n=1}^{N} (\bar{d} - d_n)^2}$$ (39)

$$d_n = \min_{g \neq n, k \in N} (\text{dis}(s_n, s_k))$$ (40)

(3) Hyper volume (HV): This metric evaluates the convergence and distribution of Pareto solutions. A higher evaluation value indicates that the search ability of the algorithm is better. The formulation of HV is shown in equation (41) [31], [32], where $\text{vol}_n$ represents the hyper volume consisted of the reference point and the $n$th solution, and $\delta$ denotes the Lebesgue measure utilized to compute the volume. In this paper, the reference point is set (1,1).

$$HV = \delta(\bigcup_{n=1}^{N} \text{vol}_n)$$ (41)

**C. PERFORMANCE COMPARISON OF DIFFERENT ALGORITHMS**

To analyze the performance of record matrix based search strategy in RMMOCS algorithm, this paper compares MOCS, NSGA-II, MOABC, MOFA, and MOACO algorithm with RMMOCS algorithm. The parameter settings of six algorithms are shown in Table 6. To avoid the randomness of the algorithm operation, each algorithm runs independently 20 times for each instance.

Table 7 shows the best results of six algorithms. As shown in Table 7, the RMMOCS algorithm can get smaller objective values at both objectives, except for two instances (instance $6 \times 30$ and instance $10 \times 50$). In addition, RMMOCS algorithm has the most Pareto solutions, i.e. 24, 21, 32, and 30.

Fig. 11 shows the best Pareto front of RMMOCS, MOCS, and NSGA-II algorithms in different instances. The Pareto front obtained by RMMOCS algorithm are slightly better than those of MOCS algorithm and have the similar performance as NSGA-II algorithm in the first instance. In the other instances, RMMOCS algorithm is superior to other two algorithms and can obtain lower multiple objective values. As the instance size increases, RMMOCS algorithm can still possess a high performance, which shows the efficiency of the search strategy based on the record matrix. The search strategy based on the optimal record matrix can enhance the local search ability of the algorithm, while the search strategy based on the history record matrix can be beneficial to the global search of the algorithm. Therefore, RMMOCS algorithm can obtain better search results than the others.

Fig. 12 shows the optimal Pareto front of RMMOCS, MOABC, MOFA, MOACO algorithms in different instances. In all instances, the quality of Pareto front obtained by RMMOCS algorithm is the best, followed by the MOABC algorithm and MOFA algorithm, and the MOACO algorithm is the worst. In fact, MOABC and MOFA algorithms generate new solutions by referencing the better solution, while MOACO algorithm does not reference the better solution in the process of producing the new solution, but only uses the local optimal solution to evaluate the quality of the new solution. It can be found that the algorithm can get better performance by utilizing more optimal solutions. Referencing the information of all optimal solutions by using the record matrix, the search ability can be enhanced for RMMOCS algorithm. The results show that for multi-objective algorithms, choosing the appropriate reference direction in the iterative process is important for the search results of the algorithm.

Overall, the quality of Pareto solutions obtained by RMMOCS algorithm is superior to other algorithms, which suggests that the search strategy based on the record matrix can effectively improve the search ability of the algorithm.

To further analyze the performance of the proposed algorithm, three multi-objective evaluation metrics (GD, SP, and HV) are utilized to compare these algorithms. The evaluation results are shown in Tables 8-10, and Fig. 13-15 are the corresponding box plots. RMMOCS algorithm provides superior
results compared to other algorithms across GD metric, except for the first instance. Meanwhile, as the instance size increases, the performance of RMMOCS algorithm is significantly improved. For SP metric, RMMOCS algorithm can obtain a lower mean value, except for the top two instances. MOCs, NSGA-II, and MOABC have the similar performance across GD and SP metrics, while MOFA and MOACO perform the worst. For HV metric, RMMOCS algorithm can obtain higher values than other algorithms, i.e. 44.88, 30.17, 20.62, and 7.64, which indicates RMMOCS algorithm has a better comprehensive performance in comparison to other algorithms across HV metric. MOCs, NSGA-II, MOABC, and MOFA have the similar performance across GD and SP metrics, and MOACO is the worst. Generally, the

| Algorithm | Parameter  | Value | Algorithm | Parameter  | Value |
|-----------|------------|-------|-----------|------------|-------|
| RMMOCS    | Population size | 100   | NSGA-II   | Population size | 100   |
|           | Archive size      | 100   |           | Archive size      | 100   |
|           | $P_c$           | 0.25  |           | Crossover probability | 0.85  |
|           | $Rou$           | 0.2   |           | Mutation probability | 0.25  |
| MOCs      | Population size | 100   | MOABC     | Population size | 100   |
|           | Archive size      | 100   |           | Archive size      | 100   |
|           | $P_c$           | 0.25  |           | Limit            | 20    |
| MOACO     | Population size | 100   | MOFA      | Population size | 100   |
|           | Archive size      | 100   |           | Archive size      | 100   |

**TABLE 7.** Best results of different algorithms (best results are in bold).

| Instance | RMMOCS | MOCS | NSGA-II |
|----------|--------|------|---------|
|          | Min $f_1$ | Min $f_2$ | Min $f_1$ | Min $f_2$ | Min $f_1$ | Min $f_2$ |
| 4x20     | 826.93   | 1399  | 24      | 840.27    | 1437    | 22      | 827.92    | 1405    | 24      |
| 6x30     | 1286.11  | 1728  | 21      | 1284.96   | 1974    | 16      | 1285.66   | 1958    | 17      |
| 8x40     | 1644.11  | 2028  | 32      | 1648.06   | 2426    | 17      | 1647.92   | 2287    | 18      |
| 10x50    | 2075.59  | 2667  | 30      | 2077.98   | 3238    | 19      | 2072.79   | 3168    | 18      |

| Instance | MOABC | MOFA |
|----------|-------|------|
| 4x20     | MOABC | MOACO|
| 6x30     | 1289.08 | 1936 |
| 8x40     | 1648.18 | 2470 |
| 10x50    | 2078.61 | 3320 |

**TABLE 8.** GD values of different algorithms (best results are in bold).

| Instance | RMMOCS | MOCS | NSGA-II | MOABC | MOFA | MOACO |
|----------|--------|------|---------|-------|------|-------|
|          | AVG   | SD   | AVG    | SD    | AVG  | SD    |
| 4x20     | 5.43  | 3.26 | 7.85   | 4.54  | 5.03 | 3.02  |
| 6x30     | 11.74 | 25.57 | 66.54  | 53.66 | 71.67 | 61.45 |
| 8x40     | 1.11  | 4.77 | 70.72  | 64.45 | 58.95 | 46.77 |
| 10x50    | 1.68  | 3.66 | 60.45  | 35.04 | 55.34 | 48.69 |

|          | AVG   | SD   | AVG    | SD    | AVG  | SD    |
| 4x20     | 46.08 | 36.90 | 44.36  | 21.74 | 35.15 | 19.99 |
| 6x30     | 110.95 | 97.62 | 155.21 | 132.23 | 121.27 | 91.28 |
| 8x40     | 54.50 | 33.00 | 87.63  | 67.52 | 92.46 | 80.20 |
| 10x50    | 52.83 | 35.67 | 129.48 | 72.14 | 76.90 | 45.15 |

**TABLE 9.** SP values of different algorithms (best results are in bold).

| Instance | RMMOCS | MOCS | NSGA-II | MOABC | MOFA | MOACO |
|----------|--------|------|---------|-------|------|-------|
|          | AVG   | SD   | AVG    | SD    | AVG  | SD    |
| 4x20     | 46.08 | 36.90 | 44.36  | 21.74 | 35.15 | 19.99 |
| 6x30     | 110.95 | 97.62 | 155.21 | 132.23 | 121.27 | 91.28 |
| 8x40     | 54.50 | 33.00 | 87.63  | 67.52 | 92.46 | 80.20 |
| 10x50    | 52.83 | 35.67 | 129.48 | 72.14 | 76.90 | 45.15 |
comprehensive performance of RMMOCS algorithm is optimal, and MOCS, NSGA-II, MOABC, and MOFA perform better than MOACO.

The important advantage of record matrix of the RMMOCS is to record useful information, i.e. the information of optimal solutions and history solutions. The information of optimal solutions can be utilized to explore the optimal solution more deeply, and the information of history solutions can be utilized to explore more unknown spaces. Thus, two search mechanisms based on the optimal record matrix and history record matrix are developed. With the help of the information reflection of the record matrix, the local and global search ability of RMMOCS can be effectively improved. Moreover, the record matrix only stores the key characteristics of the solutions, which can make the algorithm address the storing process efficiently. RMMOCS still has a comprehensive performance as the instance size increases, which proves that RMMOCS possesses the potential to solve the large-scale problems.

V. CONCLUSIONS AND FUTURE WORK

This paper investigates the sequencing problem of mixed-model assembly lines in car assembly shop. Aiming at this problem, the multi-objective optimization model of car-sequencing of mixed-model assembly lines is constructed by considering the variation of parts usage rates, variation of workstation workload, idle time, overload time, and model switching cost. Based on the characteristics of mixed-model assembly processes, a multi-objective cuckoo search algorithm based on the record matrix is proposed to solve the
FIGURE 13. Box plot of GD for different algorithms. (a) Instance 4 × 20. (b) Instance 6 × 30. (c) Instance 8 × 40. (d) Instance 10 × 50.

FIGURE 14. Box plot of SP for different algorithms. (a) Instance 4 × 20. (b) Instance 6 × 30. (c) Instance 8 × 40. (d) Instance 10 × 50.
car-sequencing problem. This algorithm adopts the record matrix to record the information of the optimal solutions and historical solutions, and this information is fully utilized to provide guidance for the generation of new solutions. Specifically, the optimal record matrix only records the information of optimal solutions, while the history record matrix records all the information of generated solutions. To further enhance the performance of the algorithm, this paper combines the proposed record matrix with the cuckoo search algorithm, i.e., the information of the optimal record matrix is utilized to make the algorithm exploit the information of several optimal solutions as much as possible, and the information of the history record matrix are employed to make the algorithm explore the unknown solution spaces. The results show that the strategies and algorithm proposed in this paper can effectively address the car-sequencing problem.

However, there are still some deficiencies in this study. The record matrix proposed in this paper does not consider the case of the association information of multiple genes. In the iterative search process based on the record matrix, the computation rule of step length needs a further study. Meanwhile, future research could explore the combination of the record matrix based search strategy and other Meta heuristics. Considering the shortcomings above, future research will include the following aspects:

1. Investigate the influence of association information of multiple genes on search performance.
2. Study the influence of step length on the performance of global search and local search.
3. Consider the combination of the record matrix and other meta-heuristic strategies.

REFERENCES

[1] N. Manavizadeh, L. Tavakoli, M. Rabbani, and F. Jolai, “A multi-objective mixed-model assembly line sequencing problem in order to minimize total costs in a make-to-order environment, considering order priority,” J. Manuf. Syst., vol. 32, no. 1, pp. 124–137, Jan. 2013.

[2] I. Moya, M. Chica, and J. Bautista, “Constructive metaheuristics for solving the car sequencing problem under uncertain partial demand,” Comput. Ind. Eng., vol. 137, Nov. 2019, Art. no. 106048.

[3] M. Yavuz and H. Ergin, “Advanced constraint propagation for the combined car sequencing and level scheduling problem,” Comput. Oper. Res., vol. 100, pp. 128–139, Dec. 2018.

[4] C. S. R. Duarte, J. V. Carvalho, and A. P. Barbosa-Póvoa, “A new heuristic for car sequencing based on an integer programming approach,” IFAC Proc. Volumes, vol. 45, no. 6, pp. 164–169, May 2012.

[5] M. Siala, E. Hebrard, and M.-J. Huguet, “A study of constraint programming heuristics for the car-sequencing problem,” Eng. Appl. Artif. Intell., vol. 38, pp. 34–44, Feb. 2015.

[6] M. Drótos and T. Kis, “Car sequencing with respect to regular expressions and linear bounds,” IFAC Proc. Volumes, vol. 47, no. 3, pp. 6174–6179, 2014.

[7] N. Boyesen and M. Zenker, “A decomposition approach for the car resequencing problem with selectivity banks,” Comput. Oper. Res., vol. 40, no. 1, pp. 98–108, Jan. 2013.

[8] H. Moradi and M. Zandieh, “An imperialist competitive algorithm for a mixed-model assembly line sequencing problem,” J. Manuf. Syst., vol. 32, no. 1, pp. 46–54, Jan. 2013.
[9] H. Mosadegh, S. M. T. Fatemi Ghomi, and G. A. Säer, “Heuristic approaches for mixed-model sequencing problem with stochastic processing times,” *Int. J. Prod. Res.*, vol. 55, no. 10, pp. 2857–2880, May 2017.

[10] U. Golle, F. Rothlauf, and N. Boysen, “Car sequencing versus mixed-model sequencing: A computational study,” *Eur. J. Oper. Res.*, vol. 237, no. 1, pp. 50–61, Aug. 2014.

[11] B. Zhang, L. Xu, and J. Zhang, “A multi-objective cellular genetic algorithm for energy-oriented balancing and sequencing problem of mixed-model assembly line,” *J. Cleaner Prod.*, vol. 244, Jan. 2020, Art. no. 118845, doi: 10.1016/j.jclepro.2019.118845.

[12] H. Sun and S. Fan, “Car sequencing for mixed-model assembly line with consideration of changeover complexity,” *J. Manuf. Syst.*, vol. 46, pp. 93–102, Jan. 2018.

[13] T. C. Lopes, C. G. S. Sikora, A. S. Michels, and L. Magatão, “An iterative decomposition for asynchronous mixed-model assembly lines: Combining balancing, sequencing, and buffer allocation,” *Int. J. Prod. Res.*, vol. 58, no. 2, pp. 615–630, Jan. 2020.

[14] M. Rauf, Z. Guan, S. Sarrafzad, J. Mumtaz, E. Shehab, M. Jahanzaib, and M. Hanif, “A smart algorithm for multi-criteria optimization of model sequencing problem in assembly line,” *Robust Comput. Integr. Manuf.*, vol. 61, Feb. 2020. Art. no. 101844, doi: 10.1016/j.rcim.2019.101844.

[15] H. Mosadegh, S. M. T. Fatemi Ghomi, and G. A. Säer, “Stochastic mixed-model assembly line sequencing problem: Mathematical modeling and Q-learning based simulated annealing hyper-heuristics,” *Eur. J. Oper. Res.*, vol. 282, no. 2, pp. 530–544, Apr. 2020.

[16] P. A. Abdul Nazar and V. Madhusudanan Pillai, “Mixed-model sequencing problem under capacity and machine idle time constraints in JIT production systems,” *Comput. Ind. Eng.*, vol. 118, pp. 226–236, Apr. 2018.

[17] B. Wang, Z. Guan, S. Ullah, X. Xu, and Z. He, “Simultaneous order scheduling and mixed-model sequencing in assemble-to-order production environment: A multi-objective hybrid artificial bee colony algorithm,” *J. Intell. Manuf.*, vol. 28, no. 2, pp. 419–436, Feb. 2017.

[18] Y. G. Zhong, X. X. Lv, and Y. Zhan, “Sequencing problem for a hull mixed-model assembly line considering manufacturing complexity,” *J. Intell. Fuzzy Syst.*, vol. 30, no. 3, pp. 1461–1473, Mar. 2016.

[19] H. Mosadegh, M. Zandieh, and S. M. T. F. Ghomi, “Simultaneous solving of balancing and sequencing problems with station-dependent assembly times for mixed-model assembly lines,” *Appl. Soft Comput.*, vol. 12, no. 4, pp. 1359–1370, Apr. 2012.

[20] X. Shao, B. Wang, Y. Rao, L. Gao, and C. Xu, “Metaheuristic approaches to sequencing mixed-model fabrication/assembly systems with two objectives,” *Int. J. Adv. Manuf. Technol.*, vol. 48, nos. 9–12, pp. 1159–1171, Jun. 2010.

[21] M. Salehi, P. Fattahi, A. Roshani, and J. Zahiri, “Multi-criteria sequencing problem in mixed-model synchronous assembly lines,” *Int. J. Adv. Manuf. Technol.*, vol. 67, nos. 1–4, pp. 983–993, Jul. 2013.

[22] I. Kucukkuk and D. Z. Zhang, “Simultaneous balancing and sequencing of mixed-model parallel two-sided assembly lines,” *Int. J. Prod. Res.*, vol. 52, no. 12, pp. 3665–3687, Jun. 2014.

[23] U. Saif, Z. Guan, W. Liu, B. Wang, and C. Zhang, “Multi-objective artificial bee colony algorithm for simultaneous sequencing and balancing of mixed model assembly line,” *Int. J. Adv. Manuf. Technol.*, vol. 75, nos. 9–12, pp. 1809–1827, Dec. 2014.

[24] X. Yang and S. Deb, “Cuckoo search via Lévy flights,” in *Proc. NABIC*, vol. 3, no. 4, pp. 257–271, Nov. 1999.

[25] E. Zitzler and L. Thiele, “Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach,” *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257–271, Nov. 1999.

[26] E. Zitzler and L. Thiele, “Multiobjective optimization using evolutionary algorithms—A comparative case study,” in *Proc. PPSN*, Amsterdam, The Netherlands, 1998, pp. 292–301.

---

**HUI WANG** was born in Nanchang, Jiangxi, China, in 1993. He received the B.S. degree in mechanical engineering from the Wuhan University of Technology, Wuhan, China, in 2015, where he is currently pursuing the Ph.D. degree in mechanical engineering.

His research interests include production planning and scheduling, digital manufacturing, and project management.

**BUYUN SHENG** was born in Hunan, China, in 1964. He received the B.S. and M.S. degrees and the Ph.D. degree in mechanical engineering from the Wuhan University of Technology, Wuhan, China, in 1985, 1988, and 2001, respectively.

He is currently a Professor with the School of Mechanical and Electronic Engineering, Wuhan University of Technology. He has published more than 100 journal articles. His research interests include the digital and intelligent manufacturing, cloud manufacturing and service, manufacturing information, and CAD/CAPP/PDM.

**QIBING LU** was born in Henan, China, in 1988. He received the B.S. and M.S. degrees and the Ph.D. degree in mechanical engineering from the Wuhan University of Technology, Wuhan, China, in 2009, 2011, and 2016, respectively.

He is currently a Lecturer with the Hubei Digital Manufacturing Key Laboratory, School of Mechanical and Electronic Engineering, Wuhan University of Technology. He has published more than ten journal articles and conference papers.

**RUPIPING LUO** was born in Daye, Hubei, China, in 1993. He received the M.S. degree in mechanical engineering from the Wuhan University of Technology, Wuhan, China, in 2015 and 2017, where he is currently pursuing the Ph.D. degree in mechanical engineering.

His research interests include PLM, workflow management, and digital manufacturing.

**GAOCUI FU** was born in Guangshui, Hubei, China, in 1989. He received the B.S. and M.S. degrees in mechanical engineering from the Wuhan University of Technology, Wuhan, China, in 2013 and 2016, respectively, where he is currently pursuing the Ph.D. degree in mechanical engineering.

His research interests include CAD/CAPP/PDM, intelligent manufacturing, and digital manufacturing.

---

**[31] E. Zitzler and L. Thiele.** “Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach,” *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257–271, Nov. 1999.

**[32] E. Zitzler and L. Thiele.** “Multiobjective optimization using evolutionary algorithms—A comparative case study,” in *Proc. PPSN*, Amsterdam, The Netherlands, 1998, pp. 292–301.