SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

A multi-objective pigeon inspired optimization algorithm for fuzzy production scheduling problem considering mould maintenance

Fu Xiaoyue[1], Chan T. S. Felix[1], Niu Ben[2*], Chung S. H. Nick[1] & Qu Ting[3]

[1]*Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Kowloon* 999077*, Hong Kong;*
[2]*College of Management, Shenzhen University, Shenzhen* 518060*, China;*
[3]*School of Electrical and Information Engineering, Jinan University, Guangzhou* 519070*, China*

Abstract In this research, the Fuzzy Production Scheduling Problem considering Mould Maintenance (FPSP-MM) is studied. The processing time and the maintenance time are represented by triangular fuzzy numbers. When tasks are executed based on the sequence provided by the fuzzy schedule, the real duration of each task needs to be known and the posteriori solution with deterministic processing times is obtained. Therefore, the concept of the schedule robustness needs to be considered for the fuzzy problem. The robustness is considered as the optimization objective except for the fuzzy makespan in this research. To optimize these two objective functions, a Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm is proposed. To extend Pigeon Inspired Optimization (PIO) algorithm from the single-objective case to the multi-objective case, non-dominated solutions are used as candidates for the leader pigeon designation and a special crowding distance is used to ensure a good distribution of solutions in the objective space and in the corresponding decision space. Furthermore, an index-based ring topology is used to manage the convergence speed. Numerical experiments on a variety of simulated scenarios show the efficiency and effectiveness of the proposed Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm by comparing it with other algorithms.

Keywords Fuzzy, Production Scheduling, Mould Maintenance, Pigeon Inspired Optimization, Multi-Objective

Citation Fu Xiaoyue, Chan T. S. Felix, Niu Ben, et al. A multi-objective pigeon inspired optimization algorithm for fuzzy production scheduling problem considering mould maintenance. Sci China Inf Sci, for review

1 Introduction

Production scheduling is to assign some tasks to limited resources so that all the constraints are satisfied, and all the objectives are achieved. There are different types of production scheduling problems. However, in most of these production scheduling problems, only the allocation of machines is considered. In some industries such as the plastic production industry and die stamping production industry, the mould is also an important resource that needs to be considered. Traditionally, resources are assumed to be always available in the whole production planning stage. However, in real situations, some machines may be unavailable because of the stochastic failures [1]. So, the maintenance on resources needs to be considered when production plans are made. The system productivity is improved when resource maintenance planning and production scheduling are integrated [2]. To integrate production scheduling

* Corresponding author (email: drniuben@gmail.com)

with multi-resource maintenance, some researchers have made great efforts. To minimize the overall makespan, Wong, Chan, and Chung (2012) [3] proposed a joint scheduling strategy to solve the integrated production scheduling with mould maintenance problem. Besides, a more complex integrated problem that contains multiple resources and maintenance tasks was considered by Wong, Chan, and Chung (2013) [4]. The results showed that the makespan was reduced significantly by the proposed jointly scheduling method. In addition, another integrated problem that each job includes multiple operations with multiple moulds was studied by Wong, Chan, and Chung (2014) [5]. Moreover, to minimize the makespan and unavailability of the machine and the mould, Wang and Liu (2015) [6] proposed a multi-objective integrated optimization method with NSGA-II adaption to solve the multi-objective parallel machine scheduling problem with flexible preventive maintenance on the machine and mould. Besides, setup time and mould maintenance were considered by Shen, Yang, Gao, et al. (2016) [7] to show the influence of the mould maintenance on production scheduling.

In the existing research about the integrated production scheduling problem with mould maintenance, all the information such as the processing time and the maintenance time is determinate, while this information is undetermined because of some human-related factors in most of the real-world manufacturing environments. Because of the uncertainty, the solutions built with the evaluated data may become antiquated during the implementation. Many studies model the uncertainty processing time as triangular fuzzy numbers and different algorithms are proposed to obtain good fuzzy schedules [8–10]. However, the exact starting times for each task are not obtained through the fuzzy schedule and solutions to the fuzzy problem should be treated as priori solutions. When tasks are executed according to the sequence provided by the fuzzy schedule, the real duration of each task needs to be known and a real executed schedule (the posteriori solution with deterministic data) is obtained. Therefore, the concept of schedule robustness needs to be considered for the fuzzy problem except for the fuzzy makespan which is the most often used objective [11, 12].

Pigeon Inspired Optimization (PIO) algorithm was firstly proposed by Duan and Qiao (2014) [13], which is a novel bio-inspired swarm intelligence optimizer mimicking the homing characteristics of pigeons. There are two main operators of the Pigeon Inspired Optimization (PIO) algorithm: map and compass operator and landmark operator. Since the Pigeon Inspired Optimization (PIO) algorithm was proposed, it has been used in many real-world applications and many new variants based on it are put forward. To achieve the target detection task for Unmanned Aerial Vehicles (UAVs) at low altitude, the hybrid model of Edge Potential Function (EPF) and Simulated Annealing Pigeon Inspired Optimization (SAPIO) algorithm were proposed by Li and Duan (2014) [14]. The robustness and effectiveness of the SAPIO algorithm are shown by a number of comparative experiments with other algorithms. Moreover, a novel Predator-Prey Pigeon-Inspired Optimization (PPPIO) [15] was proposed to solve the Uninhabited Combat Aerial Vehicle (UCAV) three-dimension path planning problem in the dynamic environment. The comparative simulation results show that the proposed PPPIO algorithm is more efficient than other algorithms for solving the problem. In addition, an orthogonal PIO algorithm [16] was suggested and employed in the training process of the Echo State Network (ESN) to obtain desired parameters. The superiority of the orthogonal PIO algorithm is shown by comparing with several existing bio-inspired optimization algorithms. Furthermore, an improved Pigeon-Inspired Optimization (PIO) algorithm [17] was utilized by converting the parameter design problem for the automatic carrier landing system to an optimization problem. A series of experiments are conducted to demonstrate the feasibility and effectiveness of the proposed method. Comparative results indicate that the proposed method is much better than other methods. In addition, a Gaussian PIO (GPIO) [18] was proposed for solving the optimal formation reconfiguration problems of multiple orbital spacecraft. The feasibility and effectiveness of the proposed Gaussian PIO (GPIO) in solving orbital spacecraft formation reconfiguration problems are verified by the comparative experiments with basic PIO and particle swarm optimization (PSO).

Furthermore, Pigeon Inspired Optimization (PIO) is also extended to solve multi-objective problems. Qiu and Duan (2015) [19] proposed a Multi-objective Pigeon Inspired Optimization (MPIO) to solve the multi-objective optimization problems in designing the parameters of brushless direct current motors. Comparative experimental results with the modified non-dominated sorting genetic algorithm are given

to show the feasibility, validity, and superiority of the proposed algorithm. Moreover, they modified the Multi-objective Pigeon-Inspired Optimization (MPIO) based on the hierarchical learning behavior in pigeon flocks and an Unmanned Aerial Vehicle (UAV) distributed flocking control algorithm based on the modified MPIO was proposed to coordinate UAVs to fly in a stable formation under complex environments. Comparison experiments with basic MPIO and a modified non-dominated sorting genetic algorithm (NSGA-II) are carried out to show the feasibility, validity, and superiority of the proposed algorithm [20].

Although there are a few studies about the production scheduling problem considering mould maintenance, none of them consider the indeterminate processing time and maintenance time while the uncertainty needs to be considered in practical application. Furthermore, when the fuzziness is considered in the integrated problem, the robustness should be considered because that there may exist some differences between the fuzzy schedule and the actually execute schedule. Moreover, as a new proposed swarm intelligent algorithm, the Pigeon Inspired Optimization (PIO) algorithm has not been applied into the production scheduling problem since it was proposed, and more efficient variants of PIO need to be explored to solve real-world problems.

Based on the research gap, this paper proposes a Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm for Fuzzy Production Scheduling Problem considering Mould Maintenance (FPSP-MM). The uncertainty is restricted to the fuzzy processing time and fuzzy maintenance time, which means that the processing time and the maintenance time can be represented by triangular fuzzy numbers. There are two objectives in this optimization problem, the fuzzy makespan, and the robustness. To extend PIO algorithms from the single-objective to the multi-objective case, non-dominated solutions are used as candidates for the leader pigeon and a special crowding distance is used to ensure a good distribution of solutions in the decision space and in the corresponding objective space. Furthermore, an index-based ring topology is used to manage the convergence speed. To evaluate the results, the Hypervolume (HV) and the Cover Rate (CR) are used as two performance indicators. In the experiments, some instances are generated by fuzzifying the benchmarks from the deterministic problem and some instances are generated randomly. To show the advantages of the proposed Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm, MOPSO and NSGA-II which are the most popular algorithms to solve multi-objective problems are used as the comparison algorithms.

The reminder of this paper is organized as follows: Section 2 describes the Fuzzy Production Scheduling Problem with Mould Maintenance (FPSP-MM) problem. Section 3 proposes the Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm. Section 4 presents the computational results acquired and shows the superiority of the Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm. Section 5 provides the conclusions and suggestions for further research.

2 Problem description

The Fuzzy Production Scheduling Problem with Mould Maintenance (FPSP-MM) problem can be described as follows: P jobs are allocated on Q injection machines and N injection moulds. Each problem is denoted as $P * Q * N$. At time zero, all jobs are well prepared, and all the machines and moulds are accessible. Each job must use the mould which is given in advance. Each job can choose the machines it uses, but not all the machines are available for all jobs. Different jobs may be performed by the same mould. Each job cannot be performed by more than one machine at a given time slot. And each machine cannot deal with more than one job at a given time slot. Each mould cannot only carry out more than one job at a given time slot. The batch size and the unit fuzzy operation time of each job are given. The total operation time of a job is the product of the unit fuzzy operation time and the batch size. The batch size of each job cannot be split and there is no interruption during the production process of a job. The unit fuzzy operation time of a job depends on the mould it uses. The unit fuzzy operation time is represented by a triangular fuzzy number. The maintenance time of resources depends on the time that the maintenance begins. The maintenance time is shorter when the maintenance is conducted earlier.

The maintenance time of a resource depends on the longest accumulated working time of a resource (the accumulated working time of a resource is a triangular fuzzy number, the longest means the third number of the triangular fuzzy number). The maintenance time may be fuzzy or crisp. Only perfect maintenance is considered, which means that after the maintenance, the condition of the resource is as good as new. In this paper, the preventive maintenance is assumed to be able to prevent all the random breakdowns. Moreover, the set-up time and the quality issue are not considered in this research. The objective is to find a good production scheduling and machine maintenance planning aiming at minimizing the makespan and maximize the robustness. To extend the determinate single-objective problem to a fuzzy multi-objective problem. Four problems need to be solved, the definition of the arithmetic operations on triangular fuzzy numbers, the fuzzy maintenance time, the robustness and the Pareto dominance relationship. These problems are given in Section 2.1-Section 2.4.

2.1 Arithmetic operations on triangular fuzzy numbers

In this paper, the unit fuzzy processing time is a triangular fuzzy number or TFN, denoted as A=(a^1, a^2, a^3),a^1 is the best processing time, a^3 is the worst processing time and a^2 is the most possible processing time. When the original deterministic model is extended to a model with uncertainty. Two difficulties need to be solved. Firstly, the arithmetic operations of addition and maximum need to be given when deterministic numbers are changed into TFNs. Secondly, the definition of minimal makespan also need to be given when the makespan is a triangular fuzzy number. According to the literature [21], for two TFNs, A=(a^1, a^2, a^3) and B=(b^1, b^2, b^3). The addition of them is defined as:

$$A + B = ((a^1 + b^1), (a^2 + b^2), (a^3 + b^3)) \tag{1}$$

The maximum operation is defined as:

$$max(A, B) = (max(a^1, b^1), max(a^2, b^2), max(a^3, b^3)) \tag{2}$$

To find the minimal makespan, three ranking criteria are given as follows:

$$C_1(A) = \frac{a^1 + 2 * a^2 + a^3}{4} \tag{3}$$

$$C_2(A) = a^2 \tag{4}$$

$$C_3(A) = a^3 - a^1 \tag{5}$$

To compare two TFNs, the values of C_1 are compared. If the values of C_1 are the same for two TFNs, then the values of C_2 are compared. If the values of C_1 and C_2 are the same for two TFNs, then the values of C_3 are compared.

2.2 Uncertain maintenance time

In the determinate model proposed by Wong, Chan,and Chung (2012) [3], the accumulated working time of a resource is defined as the resource age (the idle time is not included). A piecewise linear function is used to describe the relationship between maintenance time and resource (machine or mould) age. In practice, a mould has a higher possibility of breakdown than a machine. So, the maximum age of the machine (MA) is longer than the maximum age of the mould (NA). Maintenance has to be conducted after completion of the current job once a resource reaches its maximum age. In the uncertainty model, since the processing time is a triangular fuzzy number, the resource age is also a triangular fuzzy number. The maintenance time is decided by the worst value in the triangular fuzzy number of the resource age. The age of the machine is represented by A=(a^1, a^2, a^3) . The age of mould is represented by B=(b^1, b^2, b^3) . The relationship between the maintenance time and the resource age is shown in Table 1.

Table 1 Maintenance time based on machine/mould age

Machine age	Maintenance time	Mould age	Maintenance time
$0 < a^3 \leqslant 180$	(150,150,150)	$0 < b^3 \leqslant 120$	(150,150,150)
$180 < a^3 \leqslant 420$	(94,94,94)+$(a^1, a^2, a^3)/3$	$120 < b^3 \leqslant 280$	(94,94,94)+$(b^1, b^2, b^3)/2$
$420 < a^3 \leqslant 600$	(160,160,160)+$(a^1, a^2, a^3)/3$	$280 < b^3 \leqslant 400$	(160,160,160)+$(b^1, b^2, b^3)/2$
$600 < a^3$	(720,720,720)	$400 < b^3$	(720,720,720)

2.3 Objective measure

There are two objectives in this problem. The first is the fuzzy makespan and the second is the robustness. According to Palacios, Gonzlez-Rodrguez,Vela, et al. (2017) [22], the objective related to the fuzzy makespan C is defined as the expected value of the triangle fuzzy number:

$$E(C) = \frac{C^1 + 2 * C^2 + C^3}{4} \tag{6}$$

The ranking method based on the expected value is shown to be convenient and it is proven that the ranking result is similar to other ranking methods. It is obvious that the smaller the expected value, the better the objective value. The robustness is defined as:

$$Rob(C) = max\{(C^2 - C^1), (C^3 - C^2)\} \tag{7}$$

It measures the maximum possible difference between the makespan of the real execution and the most likely estimated makespan. It is a priori measure and the smaller the value, the better the robustness. So the objectives of this problem are shown as follows:

$$min\ E(C) \tag{8}$$

$$min\ Rob(C) \tag{9}$$

2.4 Pareto domination relationship

For the multi-objective optimization problems with two or more objectives to be optimized.

$$min\ f(x) = (f_1(x), f_2(x), ...f_m(x)) \tag{10}$$

where x=(x_1,x_2,...,x_n) is an n-dimensional decision vector, $f(x)$ is an m-dimensional objective vector. The n-dimensional space comprised of all the possible values of the decision vector x is known as the decision space, and the m-dimensional space consisting of all the possible values of the objective vector $f(x)$ is the objective space. There are many different solutions for multi-objective optimization problems and these solutions can be compared based on the Pareto dominance relationship: Given two feasible solutions x and y , solution x is said to dominate solution y if $f_i(x) \leqslant f_i(y)$,$i = 1, 2, ...m$ and there exists at least one $j \in 1, 2, ...m$ so that $f_i(x) < f_i(y)$. A solution is said to be non-dominated if it is not dominated by any other solutions. The set of all the non-dominated solutions in the decision space is called the Pareto-optimal Set (PS). The Pareto Front (PF) is the set of all the vectors in the objective space that corresponds to the PS.

3 Optimization methodology

3.1 Basic pigeon inspired optimization

Pigeon Inspired Optimization (PIO) simulates the homing behaviors of pigeons. There are two main operators in the algorithm.

(1)Map and compass operator: Through shaping the map in their brains by magnetoreception, pigeons are able to sense the earth field. Moreover, pigeons adjust the directions based on the altitude of the sun,

which has the same function as a compass. When pigeons fly to their destination, they rely less and less on the sun and magnetic particles. Each pigeon has a position X_i and a velocity V_i in a D-dimension search space. Both the positions and the velocities of the pigeons are updated in each iteration. The new position X_i and velocity V_i of pigeon i at the t_{th} iteration can be calculated by the following equations:

$$V_i(t) = V_i(t-1)e^{-Rt} + rand(X_g - X_i(t-1)) \tag{11}$$

$$X_i(t) = X_i(t-1) + V_i(t) \tag{12}$$

where $V_i(t-1)$ and $V_i(t)$ are the velocities of the pigeon i at $(t-1)_{th}$ and t_{th} iteration. $X_i(t-1)$ and $X_i(t)$ are the positions of the pigeon i at $(t-1)_{th}$ and t_{th} iteration. R is the map and compass factor, $rand$ is a random number and X_g is the current global best position, and which can be obtained by comparing all the positions among all the pigeons.

(2) Landmark operator: The pigeons will depend on landmarks that are near them when the pigeons fly close to their destination. They will fly directly to the destination if they are familiar with the landmarks. They will follow the pigeons who are familiar with the landmarks if they are far from the destination and unfamiliar to the landmarks. In the landmark operator, half of the number of pigeons is decreased in every generation which means that the pigeons that are far from the destination and unfamiliar with the landmarks will follow the pigeons that are familiar with the landmarks. Then, the pigeons close to their destination will fly to their destination quickly, which is represented by $X_C(t)$ (the center of some pigeons positions at the t_{th} iteration). The position updating rule for pigeon i at the t_{th} iteration can be given by:

$$N_p(t) = \frac{N_p(t-1)}{2} \tag{13}$$

$$X_C(t) = \frac{\sum X_i(t) \cdot fitness(X_i(t))}{N_p \sum fitness(X_i(t))} \tag{14}$$

$$X_i(t) = X_i(t-1) + rand \cdot (X_C(t) - X_i(t-1)) \tag{15}$$

where $fitness(X_i(t))$ is the quality of the pigeon i at the t_{th} iteration. For the minimum optimization problems, it is usually chosen as $fitness(X_i(t)) = \frac{1}{f_{min}(X_i(t))+\xi}$. For the maximum optimization problems, it is usually chosen as $fitness(X_i(t)) = f_{max}(X_i(t))$. For each individual pigeon, the optimal position of the Nc_{th} iteration can be denoted with X_p , and $X_p = min(X_i(1), X_i(2), ..., X_i(Nc))$.

3.2 Encoding and decoding of the pigeon

In the Multi-Objective Pigeon Inspired Optimization (MOPIO), each pigeon contains information on the job sequence (J), the corresponding machine sequence (M), machine maintenance (AM) and information on the mould maintenance (OM). In the evolution process of MOPIO, the positions values of these pigeons always fluctuate in the space of real number. Random key representation [23] and the smallest position value (SPV) rule [24] are applied to decode the positions of pigeons into a suitable scheduling solution for this problem. After decoding, the values of the J parameters are integers between 1 and P (P is the number of jobs); the values of M parameters are integers between 1 and Q (Q is the number of machines); The AM parameter is the maintenance decision on the machine, with value 0 or 1; The OM parameter is the maintenance decision on the mould, with value 0 or 1; If the relevant AM or OM is denoted as 1, the corresponding resource is maintained after finishing the job, otherwise they are not maintained. An example pigeon before and after decoding is shown in Figure 1. In this example, there are 4 jobs, 2 machines, and 2 moulds. Jobs 1, 2 can only be produced by mould 1 and Jobs 3, 4 can only be produced by Mould 2. From Figure 1, it can be seen that the value of the job sequence (J) in the original position of the pigeon is (1 0.3 0.8 1.2), and it is transferred into (2 3 1 4). The sequence (1 0.3 0.8 1.2) is ranked according to the ascending order and obtain (0.3 0.8 1 1.2). Since the number 0.3 is in the second position of the original sequence, it is decoded into 2. Since the number 0.8 is in the third position of the original sequence, it is decoded into 3. Using this method, the original positions can be decoded into a suitable scheduling solution (2 3 1 4). The value of the corresponding machine sequence

Original position of the pigeon:															
1	0.3	0.8	1.2	0.5	1.1	0.6	0.1	0.6	0.2	1	0.5	1.2	0.2	0.8	0.4
J	J	J	J	M	M	M	M	AM	AM	AM	AM	OM	OM	OM	OM
Scheduling solution after decoding:															
2	3	1	4	1	2	2	1	1	0	1	0	1	0	1	0
J	J	J	J	M	M	M	M	AM	AM	AM	AM	OM	OM	OM	OM

Figure 1 Encoding and decoding of the first example pigeon

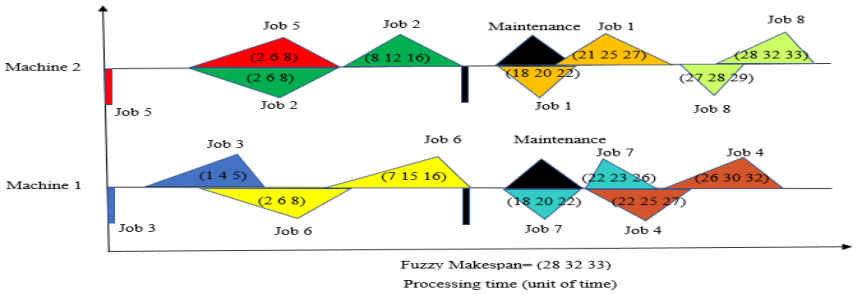

Figure 2 Machine scheduling of the second example pigeon

(M) in the original position of the pigeon is (0.5 1.1 0.6 0.1). The interval [0.1 1.1] (0.1 is the minimum and 1.1 is the maximum among all the numbers) is divided into 2 intervals, [0.1 0.6), [0.6 1.1] (there are 2 machines in this example). Since 0.5 and 0.1 are in the first interval, after decoding, the value in the relevant position is 1. Since 0.6 and 1.1 are in the second interval after decoding, the value in the relevant position is 2. So, the corresponding machine sequence (M) can be transferred into (1 2 2 1). The value of the corresponding machine maintenance sequence (AM) in the original position of the pigeon is (0.6 0.2 1 0.5) and the interval [0.2 1] (0.2 is the minimum and 1 is the maximum among all the numbers) is divided into 2 intervals, [0.2 0.6) and [0.6 1]. Since 0.2 and 0.5 are in the interval [0.2 0.6), the value in the relevant position is decoded into 0. Since 0.6 and 1 are in the interval [0.6 1], the value in the relevant position is decoded into 1. So, corresponding machine maintenance sequence (AM) is decoded into (1 0 1 0). A similar decode method can be applied to mould maintenance (OM). After decoding, it could be known that job 2 is distributed on machine 1 and machine 1 will be maintained after job 2, and the injection mould on machine 1 will also be maintained. Job 3 is allocated to machine 2, but machine 2 will not be maintained since the corresponding AM parameter is 0 and the injection mould on machine 2 will not be maintained because the corresponding OM parameter is 0.

Furthermore, we give the fuzzy scheduling charts of another example pigeon (3 5 6 2 1 7 4 8 1 2 1 2 2 1 1 2 0 0 1 1 0 0 0 0 0 0 0 1 0 1 0 0). The machine scheduling chart is shown in Figure 2 and the mould scheduling chart is shown in Figure 3. In these figures, different jobs are represented by different colours. The fuzzy number under the line is the start time of each job and the fuzzy number above the line is the end time of each job. The maintenance on the machine and the mould is represented by the black. From the Figure 2 and Figure 3, we can know that the fuzzy makespan is (28, 32, 33)

3.3 Multi-objective pigeon inspired optimization

Inspired by the multi-objective particle swarm optimizer using ring topology proposed by Yue, Qu, and Liang (2017) [25], this paper proposes a multi-objective pigeon inspired optimization algorithm using ring topology and a special non-dominated sorting method.

In the map and compass operation of MOPIO, we use the best pigeon in the neighborhood of each pigeon instead of the global best pigeon to avoid that the population convergences to a single point. So

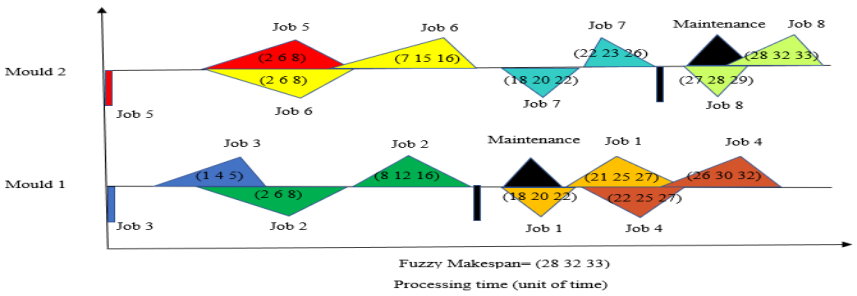

Figure 3 Mould scheduling of the second example pigeon

the equation (11) is modified into equation (16).

$$V_i(t) = V_i(t-1)e^{-Rt} + rand(X_{nbest_i} - X_i(t-1)) \tag{16}$$

where X_{nbest_i} is the best pigeon in the neighborhood of the i_{th} pigeon. Other symbols have the same meanings as the symbols in equation (11). Two archives are established including the personal best archive (PBA) and neighborhood best archive (NBA). The personal best pigeon and the neighborhood best for each pigeon are chosen from the according PBA and NBA. For the neighborhood best archive (NBA), $NBA\{i\}$ denotes the best position within the i_{th} particles neighborhood. Each neighborhood includes three particles, the i_{th} particle and its immediate neighbors on its right and left. Moreover, an index-based ring topology is used to build the neighborhood and pigeons in different neighborhoods cannot interact with each other directly. The use of the NBA promotes the formation of multiple niches by restricting the information transmission through the population. Furthermore, a special sorting scheme named non-dominated-scd-sort algorithm is used to rank the pigeons.

In the Landmark operator, the number of the pigeons is chosen as the numbers of pigeons in PBA, the fitness is defined as $fitness(X_i(t)) = \frac{1}{obj1_{min}(X_i(t))+obj2_{min}(X_i(t))+\xi}$. $obj1_{min}(X_i(t)) + obj2_{min}(X_i(t))$ is the sum of two objective values. The positions of all pi-geons are updated according to equation (13) to equation (15). And pigeons are ranked based on the non-dominated-scd-sort algorithm. When all the iterations end, best pigeons from the PBA are the final optimization solutions.

The non-dominated-scd-sort algorithm is proposed by Yue, Qu, and Liang (2017) [25]. There are two steps in this algorithm. In the first step, the pigeons are sorted according to the non-dominated sorting scheme [26]. In the second step, the special crowding distances of non-dominated pigeons are calculated. The calculation of special crowding distance also contains two steps. In the first step, the crowding distance, CD, for each pigeon in the decision space and the corresponding image in the objective space are calculated. In the second step, the crowding distances from the first step are used to assign a special crowding distance SCD for each pigeon. The SCD concept involves a max or min selection step that involves crowding metrics from the decision and objective spaces. The diversity in the solution and objective spaces are promoted simultaneously by this methodology. Finally, the non-dominated solutions are ranked in descending order according to their special crowding distances. After sorting, the first particle is the non-dominated solution with the largest special crowding distance.

The flowchart of the proposed MOPIO can be seen in Figure 4 and the details of each step are given as follows:

Step 1. Input the parameters of the fuzzy production problem with mould maintenance, including the numbers of jobs, machines, and moulds, the batch size of each job, the corresponding mould of each job, the available machines for each job, the unit fuzzy operation time of each job.

Step 2. Initialize the parameters of MOPIO, including the dimension of the solution space, the size of the population, NUM, map and compass factor R, the number of iteration Nc_1max and the number of iteration Nc_2max for two operators.

Step 3. Each pigeon is randomly allocated a posiition and a velocity. Calculate the objective values of all the pigeons. $POP_i(t)$ represents the pigeon i at the t_{th} iteration. Initialize the number of elements in the personal best archive (PBA) and the neighborhood best archive (NBA). $PBA\{i\}$ saves the best

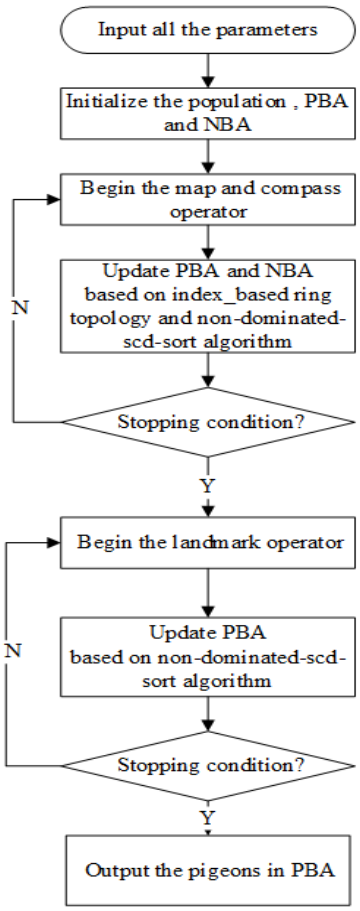

Figure 4 Flowchart of the MOPIO

position for pigeon i . $NBA\{i\}$ saves the best position in the neighborhood of the pigeon i. $PBA\{i\} = POP_i(0)$,$NBA\{i\} = PBA\{i\}$

Step 4. Begin the map and compass operator. When the iteration t_1 is smaller than the Nc_1max , for all the NUM pigeons, sort the pigeons in $PBA\{i\}$ and $NBA\{i\}$ based on the non-dominated-scd-sort algorithm. And select the first pigeon of the sorted $NBA\{i\}$ as the $nbest_i$. Update the $POP_i(t_1)$ according to Equation (16) and Equation (12). Calculate the objective values and the Special Crowding Distance(SCD) of $POP_i(t_1 + 1)$.

Step 5. Update PBA. For all the NUM pigeons, put $POP_i(t_1 + 1)$ into $PBA\{i\}$ and remove all the pigeons dominated by $POP_i(t_1 + 1)$.

Step 6. Update NBA. Using the index-based ring topology to decide the neighborhood of pigeon i , which includes three pigeon groups, the $PBA\{i\}$, $PBA\{i-1\}$ and $PBA\{i+1\}$. Particularly, if $i = 1$, the neighborhood of the pigeon i is defined as $PBA\{NUM\}$, $PBA\{1\}$ and $PBA\{2\}$; if $i = NUM$, the neighborhood is defined as $PBA\{NUM - 1\}$, $PBA\{NUM\}$ and $PBA\{1\}$. Then obtain the non-dominated pigeons based on non-dominated-scd-sort algorithm in the neighborhood and put them into the $NBA\{i\}$.

Step 7. Set $t_1 = t_1 + 1$. If the iteration t_1 is smaller than Nc_1max , return to step 4. If the iteration t_1 is bigger than Nc_1max , go to step 8.

Step 8. Begin with landmark operator. When the iteration t_2 is smaller than the Nc_2max , conduct the landmark operator. The size $N_p(1)$ is chosen as the number of pigeons in the archive PBA at the last iteration of the map and compass operation. The size $N_p(t_2)$ is chosen as the number of pigeons in the archive PBA at iteration t_2 .Then it is decreased by half in every iteration according to the equation (13), which means that half of the pigeons will follow another half of the pigeons that are

familiar with the landmark. The fitness is set as $fitness(X_i(t)) = \frac{1}{obj1_{min}(X_i(t))+obj2_{min}(X_i(t))+\xi}$. where $obj1_{min}(X_i(t)) + obj2_{min}(X_i(t))$ is the sum of two objective values. The center of the pigeons will be calculated according to the equation (14). And the positions of the pigeons will update according to equation (15).

Step 9. Update PBA. Calculate the objective values and the special crowding distance SCD of all the pigeons $POP_i(t_2+1)$. Put the new pigeons $POP_i(t_2+1)$ in the PBA and delete the pigeons dominated by $POP_i(t_2+1)$ based on the non-dominated -scd-sort algorithm.

Step 10. Set $t_2 = t_2+1$. If the iteration t_2 is smaller than Nc_2max , return to step 8. If the iteration t_2 is bigger than Nc_2max , obtain the pigeons in PBA as the final optimization results.

4 Numerical experiments

The main objective of the numerical experiments is to test the optimization performance of the proposed MOPIO algorithm. Since the problem is an extension of the problem proposed by Wong, Chan, and Chung (2012) [3]. The benchmark datasets used in this paper are generated by randomly fuzzifying the crisp benchmarks from Wong, Chan, and Chung (2012) [3]. To fuzzy a crisp benchmark dataset and generate a triangular fuzzy processing time $P_{ij} = (P_{ij}^1, P_{ij}^2, P_{ij}^3)$, according to Lei D. (2011) [27], the most plausible value P_{ij}^2 of the fuzzy processing time is equal to the value of the crisp processing time P_{ij} in the crisp datasets. The values of P_{ij}^1 and P_{ij}^2 were randomly generated from $[0.85P_{ij}, 0.95P_{ij}]$ and $[1.1P_{ij}, 1.19P_{ij}]$. The sizes of these three problems are $(30*3*5)$, $(40*6*10)$ and $(60*9*15)$. The quality of the solutions produced by the proposed MOPIO algorithm will be verified by comparing the results obtained by adapted NSGA-II [26] and MOPSO [28] . Furthermore, three more randomly generated datasets with size $(20*2*4)$, $(35*4*6)$ and $(65*8*10)$ are used as benchmarks to further illustrate the performance of the proposed MOPIO algorithm.

Numerical experiments are implemented in the Matlab environment on a personal computer with Intel (R) Core (TM) i7-6700 CPU 3.40GHz CPU.

4.1 Performance indicator

To evaluate an algorithm, the quantitative analysis is as important as the qualitative analysis. Except listing the non-dominated solutions found over a certain number of runs by different algorithms, this paper uses two performance metrics to measure the performance of different algorithms: the Hypervolume (HV) proposed by Ziter, Thiele, Laumanns , et al. (2003) [29] and the Cover Rate (CR) which is a modification from the cover rate proposed by Yue, Qu, and Liang. (2017) [25] .

HV metric is used to measure the volume of hypercube enclosed by Pareto Front A and a reference vector $r_{ref} = (r_1, r_2, ...r_n)$ with a larger value representing better performance, it is calculated as follows:

$$HV(A) = \bigcup_{a \in A} vol(a) \tag{17}$$

where $vol(a)$ is the volume of hypercube enclosed by the solution a in the Pareto Front A and the reference vector $r_{ref} = (r_1, r_2, ...r_n)$. The bigger the value, the better the algorithm.

The CR represents the overlap ratio between different Pareto Fronts obtained by different algorithms. The definition of the Cover Rate $(CR(A, B))$ is as follows:

$$CR(A, B) = (\prod_{l=1}^{n} \delta_l)^{1/2n} \tag{18}$$

$$\delta_l = \begin{cases} 1, & F_l^{max} = F_l^{min} \\ 0, & f_l^{min} \geqslant F_l^{min} \parallel f_l^{max} \leqslant F_l^{min} \\ (\frac{min(f_l^{max}, F_l^{max}) - max(f_l^{min}, F_l^{min})}{F_l^{max} - F_l^{min}})^2, & otherwise \end{cases} \tag{19}$$

Table 2 Different parameters combinations and its influence on results.

No	*Swarm size*	*Max Iteration of each operation*	*R*	*Avg(HV)*	*Sd(HV)*
1	20	100	0.01	184570	25677
2	20	200	0.2	183710	52467
3	20	400	0.4	224810	95112
4	50	100	0.2	244600	34749
5	50	200	0.4	240460	54875
6	50	400	0.01	304520	34394
7	80	100	0.4	237050	42451
8	80	200	0.01	264770	24049
9	80	400	0.2	262280	15366

where n is the dimensionality of the objective space; f_l^{max} and f_l^{min} are respectively the maximum and minimum of the l_{th} objective value obtained by algorithm A. F_l^{max} and F_l^{min} are the maximum and minimum of the l_{th} objective value obtained by algorithm B. If $CR(A,B)$ is bigger than $CR(B,A)$, it means that the scope of the Pareto Front obtained by algorithm A is larger than the scope of the Pareto Front obtained by algorithm B, which means that algorithm A is better than algorithm B in terms of the Cover Rate.

4.2 Parameter tunings

Since the parameters of the algorithm have a considerable influence on the results of the solution, we test different combinations of parameters on the medium size dataset ($40*6*10$) to decide the final parameters used for this algorithm. There are three main parameters in the proposed MOPIO, population size, the iteration (the iteration of each operation is half of the overall iteration) and the map and compass factor R . The map and compass factor R decides the influence of the previous velocity on the present velocity. The smaller the value is, the bigger the influence is. We choose the parameter combinations from the sets: Swarm size= {20, 50, 80}, Iteration ={100, 200, 400}. R= {0.01, 0.2, 0.4}. Taguchi method is used to reduce the number of experiments. Each parameter combination run ten times. The orthogonal arrays based on Taguchi methods and the average values and the standard deviation of the HV for each parameter combination are shown in Table 2. From the table 2, we can know that the larger the iteration is, the better the result is, while the larger swarm size does not mean the better the results. The smaller the R is, the better the result is. Finally, the parameter combination {swarm size=50, iteration=400 and R=0.01} is chosen as the parameter used for the following tests.

4.3 Comparison with other multi-objective algorithms

To test the performance of the proposed MOPIO, this research compares it with other two well-known multi-objective algorithms: adapted NSGA-II [26] and MOPSO [28] . For unbiased comparison, the swarm size is set as 50, the iteration is set as 400 for these three algorithms. The parameters related to the MOPSO and NSGA-II are the same as the parameters in the original literature. The R is set as 0.01 for MOPIO. The reference vector for the calculation of HV is set as (5200, 400). Each algorithm runs ten times. For the fuzzed benchmark datasets, the Pareto Fronts of three algorithms are shown in Figure 5-Figure 7. The comparison results of the performance indicators (the Hypervolume (HV) and the Cover Rate (CR)) are shown in Table 3 and Table 4. $Avg(HV)$ is the average value of the HV for the 10 runs. $SD(HV)$ is the standard deviation of the HV for the 10 runs. $Avg(CR)$ is the average value of the CR for the 10 runs. $SD(CR)$ is the standard deviation of the CR for the 10 runs.

Furthermore, three more instances are produced randomly to better illustrate the performance of the propose MOPIO algorithm. The size of these three problems are $(20*2*4)$, $(35*4*6)$ and $(65*8*10)$. The most plausible value P_{ij}^2 of the fuzzy processing time is produced randomly between 30 and 55 units of time. The values of P_{ij}^1 and P_{ij}^3 were randomly generated from $[0.85P_{ij}^2, 0.95P_{ij}^2]$ and $[1.1P_{ij}^2, 1.19P_{ij}^2]$. The batch size of the jobs is produced randomly between 2 and 6 units. The comparison results of the performance indicators are shown in Table 5 and Table 6.

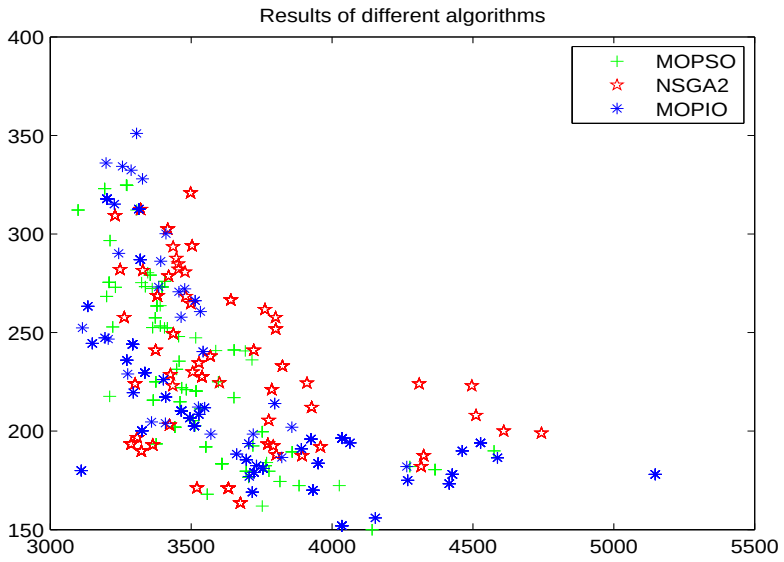

Figure 5 Pareto fronts of the fuzzed benchmark $(30 * 3 * 5)$

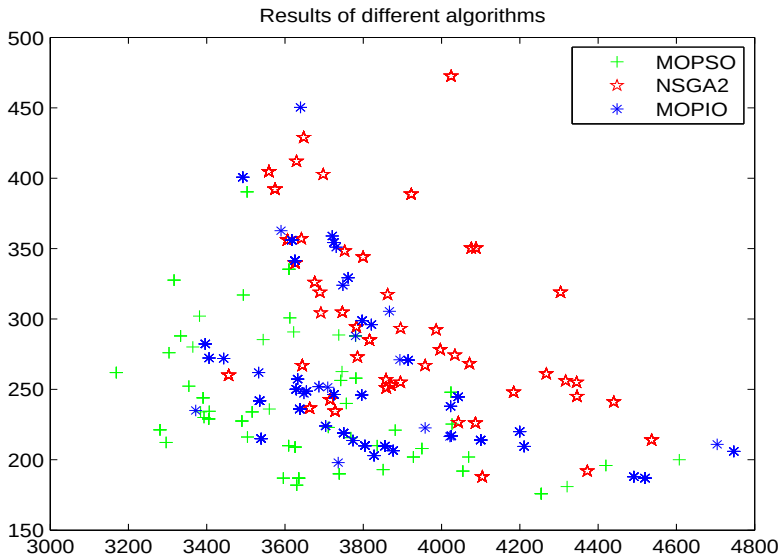

Figure 6 Pareto fronts of the fuzzed benchmark $(40 * 6 * 10)$

Table 3 HV Comparison results of the fuzzed benchmarks.

		$MOPIO$	$NSGA-II$	$MOPSO$
$30 * 3 * 5$	$Avg(HV)$	410140	369320	394800
	$Sd(HV)$	35771	38432	34513
$40 * 6 * 10$	$Avg(HV)$	296680	227000	340670
	$Sd(HV)$	33643	36029	42257
$60 * 9 * 15$	$Avg(HV)$	260000	198240	225760
	$Sd(HV)$	66773	60707	45271

From the Figure 5, it can be known that the Pareto Front of the dataset $(30*3*5)$ by MOPIO is in the left bottom of the coordinate system compared with the Pareto Fronts by MOPSO and NSGA-II. From the Table 3, we can see that for the dataset $(30*3*5)$, the value of $Avg(HV)$ for the Pareto Front by MOPIO

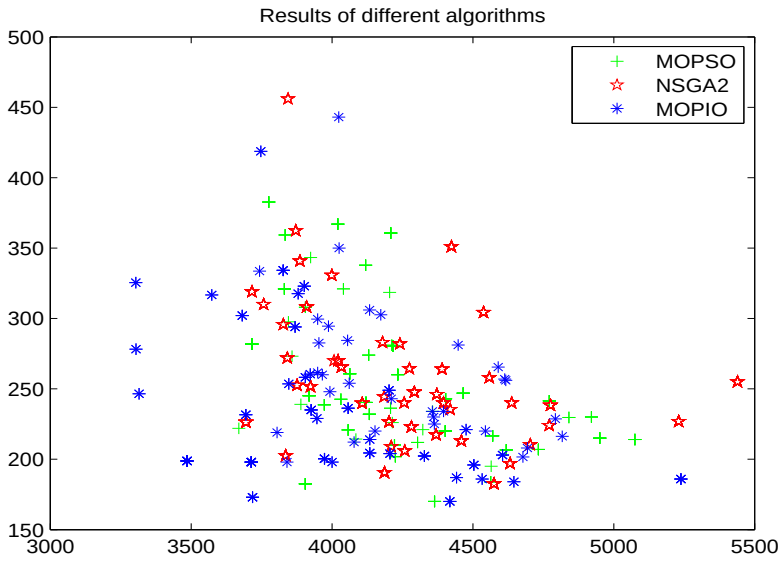

Figure 7 Pareto fronts of the fuzzed benchmark $(60 * 9 * 15)$

Table 4 CR Comparison results of the fuzzed benchmarks.

		MOPIO vs NSGA-II	NSGA-II vs MOPIO	MOPIO vs MOPSO	MOPSO vs MOPIO
$30 * 3 * 5$	$Avg(CR)$	0.7019	0.3904	0.7762	0.6362
	$Sd(CR)$	0.3260	0.2850	0.3016	0.2784
$40 * 6 * 10$	$Avg(CR)$	0.5584	0.4338	0.5456	0.5219
	$Sd(CR)$	0.2912	0.2627	0.2530	0.3083
$60 * 9 * 15$	$Avg(CR)$	0.5519	0.4986	0.71	0.5342
	$Sd(CR)$	0.334	0.3163	0.2967	0.34

Table 5 HV Comparison results of the random benchmarks.

		$MOPIO$	$NSGA-II$	$MOPSO$
$20 * 2 * 4$	$Avg(HV)$	370120	259960	420310
	$Sd(HV)$	42687	52709	44402
$35 * 4 * 6$	$Avg(HV)$	257290	115370	117340
	$Sd(HV)$	15086	26714	23554
$65 * 8 * 10$	$Avg(HV)$	357070	210300	337900
	$Sd(HV)$	24190	38107	36947

Table 6 CR Comparison results of the random benchmarks.

		MOPIO vs NSGA-II	NSGA-II vs MOPIO	MOPIO vs MOPSO	MOPSO vs MOPIO
$20 * 2 * 4$	$Avg(CR)$	0.6842	0.4734	0.7957	0.7018
	$Sd(CR)$	0.3189	0.2486	0.1846	0.2225
$35 * 4 * 6$	$Avg(CR)$	0.8664	0.2704	0.5145	0.5269
	$Sd(CR)$	0.1158	0.2374	0.4783	0.4566
$65 * 8 * 10$	$Avg(CR)$	0.7290	0.3524	0.5578	0.5319
	$Sd(CR)$	0.3821	0.3453	0.3433	0.3339

is bigger than the values of $Avg(HV)$ for the Pareto Fronts by MOPSO and NSGA-II. Furthermore, the boundary of the Pareto Front by MOPIO is larger than the Pareto Fronts by MOPSO and NSGA-II. From the Table 4, we can see that the value of the $Avg(CR(MOPIO, NSGA-II))$ is bigger than the value of the $Avg(CR(NSGA-II, MOPIO))$ and the value of the $Avg(CR(MOPIO, MOPSO))$ is bigger than the value of the $Avg(CR(MOPSO, MOPIO))$. So we can conclude that for the dataset $(30 * 3 * 5)$, the

quality of the Pareto Front by MOPIO is better than the Pareto Fronts by MOPSO and NSGA-II. For the dataset $(60 * 9 * 15)$, by comparing the location of Pareto Fronts in the Figure 7 and the values for dataset $(60 * 9 * 15)$ in Table 3 and Table 4, we can also conclude that for the dataset $(60 * 9 * 15)$, the quality of the Pareto Front by MOPIO is better than the Pareto Fronts by MOPSO and NSGA-II based on the two performance indicators.

From the Figure 6, we can see that although the Pareto Front by MOPSO is in the left bottom of the coordinate system compared with the Pareto Fronts by MOPIO and NSGA-II, the boundary of the Pareto Front by MOPIO is larger than the Pareto Fronts by MOPSO and NSGA-II. Turning back to the values in Table 3 and Table 4, we can see that for the dataset (40*6*10), the value of $Avg(HV)$ of the Pareto Front by MOPSO is bigger than the values of Pareto Fronts by MOPIO and NSGA-II. But the value of the $Avg(CR(MOPIO, MOPSO))$ is bigger than the value of the $Avg(CR(MOPSO, MOPIO))$ and the value of the $Avg(CR(MOPIO, NSGA-II))$ is bigger than the value of the $Avg(CR(NSGA-II, MOPIO))$.

From Table 5 and Table 6, it can be known that the value of the $Avg(HV)$ for MOPIO is better than the values of $Avg(HV)$ for MOPSO and NSGA-II for the bigger datasets which are randomly generated. But for the small dataset, the value of the $Avg(HV)$ for MOPSO is better than the value of the $Avg(HV)$ for MOPIO and NSGA-II. For these three datasets, the value of the $Avg(CR(MOPIO, MOPSO))$ is bigger than the value of the $Avg(CR(MOPSO, MOPIO))$ and the value of the $Avg(CR(MOPIO, NSGA-II))$ is bigger than the value of the $Avg(CR(NSGA-II, MOPIO))$.

After analyzing the Pareto Fronts of different algorithms in different aspects, we can conclude that the Pareto Fronts obtained by MOPIO always have a better Cover Rate (CR) compared with MOPSO and NSGA-II. For most of the datasets, the Hypervolume (HV) of the solutions obtained by MOPIO is better than MOPSO and NSGA-II. That is because that the mechanism of the index-based ring topology used in the MOPIO improves the diversity of the solutions and the non-dominated scd sort algorithm applied in the MOPIO to sort the pigeons helps to choose the pigeons with better performance and distribution in every iteration.

5 Conclusions

In this research, the Fuzzy Production Scheduling Problem considering Mould Maintenance (FPSM-MM) is studied. The processing time and the maintenance time are represented by triangular fuzzy numbers. Two objectives are optimized including the fuzzy makespan and the robustness. A Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm is proposed to solve this multi-objective fuzzy problem. To extend the basic Pigeon Inspired Optimization (PIO) algorithm from the single-objective case to the multi-objective case, a special non-dominated sorting method is used to obtain solutions that used as candidates for the leader pigeon and a good distribution of solutions in the objective space and in the corresponding decision space is guaranteed. Moreover, we make each pigeon exchange information with its closed neighbors instead of the global best pigeon with the help of index-based ring topology. The diversity of the population is improved by forming more niches. Furthermore, a series of experiments on the fuzzified benchmarks from existing literature and some randomly generated instances show the efficiency and effectiveness of the proposed Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm by comparing it with two well-known multi-objective algorithms.

In this research, all the jobs are well prepared at the beginning time. However, in the real cases, some new jobs may come during the production process. When the new coming tasks are considered, the original schedule needs to be adjusted to obtain better solutions. In the future, a new algorithm based on Pigeon Inspired Optimization (PIO) will be proposed to solve this problem. Furthermore, more mechanisms will be proposed to improve the effectiveness of the proposed Multi-Objective Pigeon Inspired Optimization (MOPIO) algorithm

Acknowledgements The work described in this paper was supported by grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. PolyU 15201414); The Natural

Science Foundation of China (Grant No. 71471158, 71571120, 71271140); a grant from the Research Committee of The Hong Kong Polytechnic University under student ac-count code RUKH; Project Supported by Guangdong Province Higher Vocational Colleges and Schools Pearl River Scholar Funded Scheme 2016.

References

1 Rajkumar, M., Asokan, P.,Vamsikrishna, V. A GRASP algorithm for flexible job-shop scheduling with maintenance constraints. International Journal of Production Research, 2010, 48(22): 6821-6836.
2 Berrichi, A., Yalaoui, F., Amodeo, L., et al. Bi-Objective Ant Colony Optimi-zation approach to optimize production and maintenance scheduling. Computers and Operations Research, 2010, 37(9): 1584-1596.
3 Wong, C. S., Chan, F. T. S.,Chung, S. H. A genetic algorithm approach for production scheduling with mould maintenance consideration. International Journal of Production Re-search, 2012, 50(20): 5683-5697.
4 Wong, C. S., Chan, F. T. S., Chung, S. H. A joint production scheduling approach con-sidering multiple resources and preventive maintenance tasks. International Journal of Pro-duction Research, 2013, 51(3): 883-896.
5 Wong, C. S., Chan, F. T. S.,Chung, S. H. Decision-making on multi-mould mainte-nance in production scheduling. International Journal of Production Research, 2014, 52(19): 5640-5655.
6 Wang, S. J., Liu, M. Multi-objective optimization of parallel machine schedul-ing integrated with multi-resources preventive maintenance planning. Journal of Manufac-turing Systems, 2015, 37: 182-192.
7 Shen, L., Yang, H. B., Gao, S.,et al. Production Scheduling with Mould Mainte-nance in Flow Shop. In P. Yarlagadda (Ed.), Proceedings of the 2016 4th International Con-ference on Sensors, Mechatronics and Automation. Zhuhai, China.
8 Sakawa M, Mori T. An efficient genetic algorithm for job shop scheduling problems with fuzzy processing time and fuzzy due date. Computers and industrial engineering 1999, 36: 325-341.
9 Arik, O. A.,Toksari, M. D. Multi-objective fuzzy parallel machine scheduling problems under fuzzy job deterioration and learning effects. International Journal of Production Re-search, 2018, 56(7): 2488-2505.
10 Jamrus, T., Chien, C. F., Gen, M., et al. Hybrid Particle Swarm Optimization Combined With Genetic Operators for Flexible Job-Shop Scheduling Under Uncertain Pro-cessing Time for Semiconductor Manufacturing. IEEE Transactions on Semiconductor Man-ufacturing, 2018, 31(1): 32-41.
11 Palacios, J. J., Gonzlez-Rodrguez, I., Vela, C. R.,et al. Robust multiobjective optimisation for fuzzy job shop problems. Applied Soft Computing, 2017, 56: 604-616.
12 Xiong, J., Xing, L. N., Chen, Y. W. Robust scheduling for multi-objective flexible job-shop problems with random machine breakdowns. International Journal of Production Economics, 2013, 141(1): 112-126.
13 Duan, H., Qiao, P. Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning. International Journal of Intelligent Computing and Cybernetics, 2014, 7(1): 24-37.
14 Li, C., Duan, H. Target detection approach for UAVs via improved pigeon-inspired optimization and edge potential function. Aerospace Science and Technology, 2014, 39: 352-360.
15 Zhang, B.,Duan, H. Three-dimensional path planning for uninhabited combat aerial vehicle based on predator-prey pigeon-inspired optimization in dynamic environment. IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB), 2017, 14(1): 97-107.
16 Duan, H.,Wang, X. Echo State Networks With Orthogonal Pigeon-Inspired Optimi-zation for Image Restoration. IEEE Transactions on Neural Networks and Learning Systems, 2016, 27(11): 2413-2425.
17 Deng, Y., Duan, H. Control parameter design for automatic carrier landing system via pigeon-inspired optimization. Nonlinear Dynamics, 2016, 85(1): 97-106.
18 Zhang, S., Duan, H. Gaussian pigeon-inspired optimization approach to orbital space-craft formation reconfiguration. Chinese Journal of Aeronautics, 2015, 28(1): 200-205.
19 Qiu, H., Duan, H. Multi-objective pigeon-inspired optimization for brushless direct current motor parameter design. Science China Technological Sciences, 2015, 58(11): 1915-1923.
20 Qiu, H., Duan, H. A multi-objective pigeon-inspired optimization approach to UAV distributed flocking among obstacles. Information Sciences. 2018.
21 Fortemps, P. Job shop scheduling with imprecise durations: a fuzzy approach. IEEE Transactions on Fuzzy Systems, 1997, 5(4): 557-569.
22 Palacios, J. J., Gonzlez-Rodrguez, I., Vela, C. R., et al. Robust multiobjective optimisation for fuzzy job shop problems. Applied Soft Computing, 2017,56: 604-616.
23 Bean, J. C. Genetic Algorithms and Random Keys for Sequencing and Optimization. ORSA Journal on Computing, 1994, 6(2): 154-160.
24 Tasgetiren, M. F., Liang, Y.C., Sevkli, M., et al. A particle swarm opti-mization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem. European Journal of Operational Research, 2007, 177(3): 1930-1947.
25 Yue, C., Qu, B.,Liang, J. A multi-objective particle swarm optimizer using ring to-pology for solving multimodal multi-objective problems. IEEE Transactions on Evolutionary Computation. 2017, DOI: 10.1109/TEVC.2017.2754271
26 Deb, K., Pratap, A., Agarwal, S., et al. A fast and elitist multi-objective genetic algorithm: NSGA-II. IEEE transactions on evolutionary computation, 2002, 6(2): 182-197.
27 Lei, D. Scheduling fuzzy job shop with preventive maintenance through swarm-based neighborhood search. The

International Journal of Advanced Manufacturing Technology, 2011, 54(9-12): 1121-1128.

28 Coello, C. A. C., Pulido, G. T., Lechuga, M. S. Handling multiple objectives with par-ticle swarm optimization. IEEE Transactions on evolutionary computation, 2004, 8(3): 256-279.

29 Zitzler, E., Thiele, L., Laumanns, M., et al. Performance assessment of multiobjective optimizers: An analysis and review. IEEE Transactions on evolutionary computation, 2003, 7(2): 117-132.