



A multiobjective memetic algorithm for integrated process planning and scheduling problem in distributed heterogeneous manufacturing systems

Qihao Liu¹ · Xinyu Li¹ · Liang Gao¹ · Guangchen Wang¹

Received: 14 July 2021 / Accepted: 4 April 2022 / Published online: 9 May 2022
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2022

Abstract

With the deepening of economic globalization and the development of the manufacturing industry, distributed manufacturing patterns have become a popular topic in current production research. In the background of distributed shop scheduling, process planning problems in different factories are considered integrally with scheduling problems to utilize the heterogeneous machining resources of distributed factories. To address actual production problems more concretely, this paper investigates the multiobjective distributed integrated process planning and scheduling (MODIPPS) problem to minimize makespan, maximum machine load, and total machine load, and it establishes a mixed-integer linear programming (MILP) model. In addition, by designing a new encoding method based on the OR-nodes of the process network graph, this paper proposes a multiobjective memetic algorithm (MOMA) to solve the problem. The proposed MOMA can guarantee the feasibility of individuals by several specially designed genetic operators so that the process precedence constraints in the network graph are satisfied in the whole algorithm period. Furthermore, the algorithm introduces a simulated annealing (SA) mechanism to avoid falling into a local optimum by accepting relatively poor individuals with a certain probability. Finally, through comparison experiments on benchmarks, the proposed method shows sufficient effectiveness and superiority in solving MODIPPS problems compared with existing classic multiobjective optimization algorithms.

Keywords Multiobjective optimization · Memetic algorithm · Distributed scheduling · Heterogeneous process

1 Introduction

In the background of economic globalization [11], the demand for individualization of orders has greatly increased, which has led to a diversification of products [12, 27, 33]. Enterprises are developing in the direction of mass customization. It is becoming increasingly difficult for traditional manufacturing and management patterns to meet the development requirements of modern enterprises. Therefore, the manufacturing mode has begun to evolve from the traditional single workshop to multiple distributed workshops [25, 28]. Distributed manufacturing has gradually become an important trend and has been successfully applied in

many important industries, such as aircraft, aerospace, and electronic products [9, 26, 40]. Focusing on distributed manufacturing systems, distributed shop scheduling studies plans for distributing jobs among enterprises, factories, and shops and the production schemes within each enterprise, factory, or shop to optimize production indices [34]. Therefore, it is critical to generate production schemes based on the concrete conditions of different factories. By rationally allocating production tasks across factories, managers can better hedge the risks associated with single production and reduce the corresponding production costs and product delivery cycles [25, 31].

Distributed manufacturing patterns have become a typical phenomenon in industry [24]. Current research related to distributed manufacturing focuses on distributed flow shop problems (DFSPs) or distributed job shop problems (DJSPs). The structures of factories are mostly homogeneous [19]. However, in fact, due to the different production capabilities of different factories, they can provide completely different alternative process plans for the same part. Treating

✉ Xinyu Li
lixinyu@mail.hust.edu.cn

¹ State Key Laboratory of Digital Manufacturing Equipment and Technology, School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China

each factory as a simple homogeneous job shop environment constitutes neglect and waste of flexible manufacturing capability. Therefore, considering heterogeneous process flexibility in distributed manufacturing, i.e., distributed integrated process planning and scheduling (DIPPS) problems, is more suitable for actual production environments. Compared with the DJSP or distributed FJSP (DFJSP), the DIPPS problem is more general and has a more complex solution space structure, which also means that it is more relevant to actual production requirements and more valuable for research.

By taking full advantage of the complementarity of process planning and shop scheduling, their integration can greatly facilitate the development of an intelligent manufacturing system [1]. In the last decade, the integrated process planning and scheduling (IPPS) problem has become a research hotspot in the manufacturing system area. It is also an NP-hard problem and is more complicated than the job shop scheduling problem [29]. Process planning has a large number of manufacturing flexibilities that can provide more choices for production but makes it difficult to obtain the optimal process plan [14]. These complex constraints and flexibilities bring great difficulty for modeling and solving this problem. Traditionally, process planning and shop scheduling are carried out sequentially. Scheduling the jobs whose process routes are predetermined means that the process planning system cannot provide flexible support for the scheduling system. This can lead to some production problems, such as unbalanced machine loads, bottlenecked resources, and conflicting optimization objectives in the distributed manufacturing system [20]. Therefore, it is essential to carry out research on IPPS in distributed manufacturing systems.

There are few existing publications on DIPPS, and their innovations have mainly concerned the design of chromosome representation methods for genetic algorithms (GAs) [13, 35, 36]. These studies were very creative, with novel ideas for subsequent DIPPS research. However, the above research did not greatly improve the overall framework of the algorithm. Their encoding methods were carried out by encoding the process routes generated from the network graph rather than directly encoding the process network itself. This indirect intermediate method can allow the algorithms to miss some potential optimal solutions. In addition, the work of generating process routes from various networks with complex topological relationships can consume many human and computational resources and reduce the efficiency of production management. Most importantly, the above research assumed that the number of process routes was enumerable and limited, but it was not. The jobs used as examples in these publications have only 3–5 alternative process routes [13], while the actual process planning (PP) problem has been proven to be NP-hard in the literature [17]. Therefore, it cannot be simply represented by a limited num-

ber (3–5) of routes, and such simplification is inappropriate. In this paper, a mathematical model of DIPPS based on a network graph is developed, and a new encoding and decoding approach is designed. This approach directly codifies the selection state of the OR-nodes in the chromosomes, which can enable more kinds of representations for process routes and the effective reconstruction of the solution space.

Most production problems are multiobjective, while all the current studies of DIPPS problems are single-objective [13, 35, 36, 38]. Aiming to solve multiobjective problems in actual production and to provide high-quality alternative schemes for production management decision-makers, this paper investigates the multiobjective DIPPS (MOIPPS) problem to minimize makespan, total machine workload, and maximum machine workload. Common handling methods for multiobjective problems include the weight-based method and the Pareto-based method. Because it is difficult to trade off weight coefficients between objectives, the Pareto-based method has become mainstream in current multiobjective optimization research. As a typical representative of many meta-heuristics, the GA has always been one of the main methods of solving scheduling problems. However, the traditional GA is not as powerful in local search as it is in global search. The memetic algorithm (MA) is a combination of global search based on population and local heuristic search based on individuals; thus, it has a good balance of exploration ability and exploitation ability. MA has been successfully applied in solving IPPS problems and has achieved a certain degree of success. To improve the performance of traditional MA, a new encoding method for chromosome representation is proposed and integrated into the MA framework, and a multiobjective MA (MOMA) is further proposed to solve MODIPPS problems. For the local search component of the MOMA, a simulated annealing (SA) mechanism is introduced to receive relatively worse individuals with a certain probability in the neighborhood of the current solution, which can prevent the algorithm from falling into a local optimum prematurely and can balance the global search capability and local search capability. For fairness of comparison, numerical experiments are conducted on open benchmarks. The main contributions of this paper are as follows:

1. In this paper, the MODIPPS problem is studied for the first time, and a mixed-integer linear programming (MILP) mathematical model for the multiobjective problem is established based on heterogeneous process network graphs.
2. The SA mechanism is introduced to prevent the MOMA from falling into a local optimum by receiving relatively worse individuals with a certain probability, balancing the global search ability and local search ability of the algorithm.

3. The proposed MOMA can obtain a better Pareto solution set. Comparison experiments on the benchmarks show sufficient effectiveness and superiority in solving MODIPPS problems compared with existing classic multiobjective optimization algorithms.

The structure of the paper is arranged as follows: Sect. 2 reviews the relevant literature. Section 3 introduces the MODIPPS problem and the mathematical model. Section 4 presents the MOMA and the SA mechanism. Section 5 describes the numerical experiments. Section 6 concludes the research.

2 Literature review

2.1 Distributed shop scheduling problem

Jia et al. [5] proposed distributed manufacturing to reduce product cost and risk and proposed an improved genetic algorithm to solve the DJSP. Naderi et al. [23] studied the DFSP and established six different MILP models and then proved that the distributed permutation flow shop scheduling problem (DPFSP) is an NP-hard problem. To meet the requirements of different manufacturing modes, Ying et al. [34] established a MILP model for the distributed hybrid flow shop scheduling problem (DHFSP) for the first time and designed an iterative greedy (IG) algorithm. As an extension of the DFSP, optional flexible constraints were added to the DHFSP. Zhao et al. [40] proposed an improved differential evolution algorithm for the DFSP with blocking to minimize the maximum completion time. Giovanni et al. [3] researched the DFJSP, which can be regarded as an extension of the DJSP. Based on the above work, Wu et al. [33] investigated the distributed assembly flexible job shop scheduling problem and proposed a MILP model. To minimize tardiness and cost, a hybrid multiobjective algorithm of the differential evolution (DE) algorithm and SA was designed to solve the problem. Li et al. [7] first studied the distributed heterogeneous no-wait flow shop scheduling problem and proposed a discrete artificial bee colony algorithm (DABC) to solve this problem. For the distributed heterogeneous permutation flow shop scheduling problem, to minimize the maximum completion time, negative social impact, and energy consumption, Lu et al. [18] proposed a knowledge-based multiobjective memetic optimization algorithm (KMMOA).

In summary, the current distributed scheduling research has mainly focused on the basic flow shop scheduling problem (FSP), permutation flow shop scheduling problem, and JSP. Moreover, the environment of most factories is homogeneous. To better serve practical production, it is necessary to study distributed shop

scheduling problems with heterogeneous process flexibilities.

2.2 Multi-objective integrated process planning and scheduling problem

In the IPPS problem, process planning is often considered based on a classical scheduling problem. A process planning problem can be viewed as a constrained traveling salesman problem (TSP), which is NP-hard. Similarly, the JSP has also been proved to be NP-hard. As a problem integrating the above two, the IPPS problem is much more difficult to solve. Li et al. [8] proposed a game theory-based approach to investigate the multiobjective IPPS problem, and the Nash equilibrium in the game theory-based approach has been used to address multiple objectives. Mohammadi et al. [20] proposed a hybrid multiobjective simulated annealing algorithm (MOHSA) by fully utilizing the capability of exploration search and fast convergence. Considering IPPS as a multiobjective optimization problem in reconfigurable manufacturing settings, Mohapatra et al. [21] applied the nondominated sorting genetic algorithm-II (NSGA-II) to take into account the computational intractability of the problem. Li et al. [10] modified the solution representation with a multilayer structure and applied NSGA-II to solve MOIPPS problems. Jin et al. [6] proposed a multiobjective memetic algorithm to solve the MOIPPS problem. Considering the makespan, critical machine workload, and machine total workload as the objective functions, Shokouhi et al. [29] tried to solve the IPPS problem with the GA in a weighted-sum form. Zhao et al. [39] proposed a two-generation Pareto ant colony algorithm for multiobjective job shop scheduling problems (MOJSP) with alternative process plans and unrelated parallel machines. Aiming at saving energy and reducing carbon emissions, Zhang et al. [37] took advantage of the integration between process planning and scheduling to achieve energy savings by using a hierarchical multistrategy genetic algorithm based on nondominated sorting. For a real-world case from a battery packaging machinery workshop, Wen et al. [32] proposed a two-stage solution method based on NSGA-II for green MOIPPS problems. The current status of research related to MOIPPS is arranged chronologically in Table 1. The research objectives focus on the makespan, cost, workload, energy consumption, etc. Numerous algorithms have been proposed by researchers to solve this problem, with notable results. In this paper, the framework is also based on an evolutionary algorithm, the memetic algorithm, which incorporates local search to balance the search ability of the algorithm.

Table 1 Publications on the MOIPPS problem

Citations	Optimization objectives	Journals
[32]	Makespan, total carbon emission, and total tardiness	Swarm and Evolutionary Computation
[37]	Total energy consumption, makespan, and peak power	Energies
[39]	Maximum completion time, tardiness penalty costs, total production workload, total machine workload deviation, total production costs	Journal of Intelligent Manufacturing
[29]	Makespan, critical machine workload, machines total workload	Production and Manufacturing Research
[6]	Makespan, Maximum machine workload, Total workload of machines	International Journal of Advanced Manufacturing Technology
[10]	Makespan, cost, idle time, the maximum machine workload	China Mechanical Engineering
[22]	Makespan, cost (fixed cost and operating cost), idle time	International Journal of Production Research
[21]	Machining cost, makespan, maximization of machine utilization	International Journal of Production Research
[8]	Makespan, the maximum working time, the total workload of machines	Expert Systems with Applications
[20]	Preparation times, total tardiness	International Journal of Production Research

3 Problem formulation for MODIPPS

3.1 DIPPS problem and the network graph

In the DIPPS problem, each distributed factory can provide a process network graph for a job *i*, and the concrete process route of job *i* needs to be summarized from the network graph. As shown in Fig. 1, the network graph is composed of five types of nodes: the start node, which is virtual and represents the start of the job’s production process; the end node, which is also virtual and indicates the end of the production process; intermediate nodes, implying operations; and OR-nodes, combined with JOIN-nodes, representing process flexibilities. An intermediate node contains three types of information: the operation number in a solid circle, the optional machine number in {}, and the corresponding processing time in []. The nodes in the network are connected by arrows indicating the precedence relationship constraints between operations. Only one of

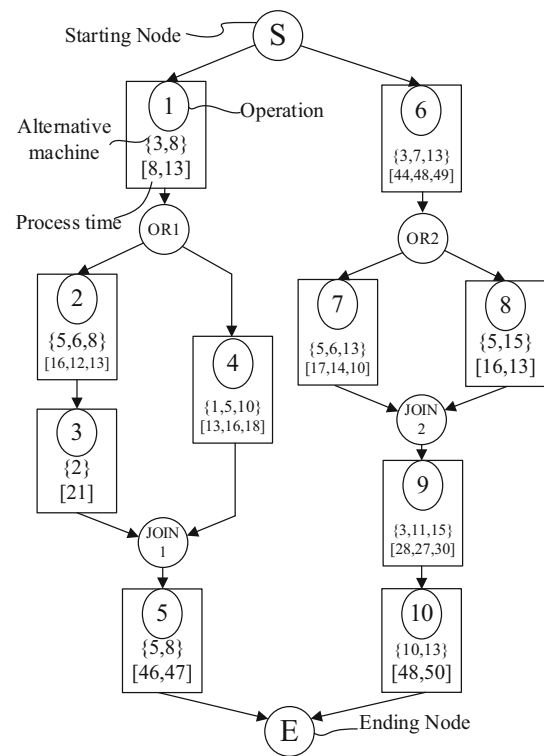


Fig. 1 An example of a network graph

the process paths between an OR-node and the corresponding JOIN-node is chosen. The operations on the chosen paths with other operations compose a feasible operation combination for the part. In addition, only one machine is selected as the platform to process the corresponding operation: 1(3)–6(7)–4(5)–7(13)–5(8)–9(3)–10(13). The numbers outside the parentheses indicate operations, and those inside are the machines. Since different jobs have different process network graphs, the number of alternative process routes may be great.

3.2 MILP mathematical model

For preliminary knowledge of the mathematical model, the details can be found in our previous work [15]. This MOIPPS problem is established based on the precedence relationship modeling idea, and the process network graph and topology analysis were discussed in detail in the previous publication [14].

The model is established based on the following assumptions:

- (1) A job can only be assigned to one factory;
- (2) A machine can only process one operation at a time, and the processing procedure cannot be interrupted;
- (3) At the beginning, all jobs and machines are already available;

(4) The transportation time of the jobs from each factory to the destination varies.

Sets, subscripts, and notation

F	The set of factories
f	Factories, $1 \leq f \leq F $
N	The job set
J_i^f	The operation set of job i if it is assigned to factory f
R_i^f	The OR-node set of job i 's process network in factory f
M^f	The machine set in factory f
O_{ij}^f	Operation j of job i if it is assigned to factory f
i, i'	Jobs, $1 \leq i \leq N $
j, j'	Operations, $1 \leq j \leq J_i^f $
k, k'	Machines, $1 \leq k \leq M^f $
r, r'	OR-nodes, $1 \leq r \leq R_i^f $
l, l'	Links
Parameters	
$U_{ijj'}^f$	1 if O_{ij}^f should be processed before $O_{ij'}^f$, according to the precedence relationship represented by job i 's network provided by factory f ; 0 otherwise
P_{ijk}^f	The processing time of O_{ij}^f on the k th machine of set M^f
W_{ijrl}^f	1 if O_{ij}^f is located at the r th OR-node's l th link in the network provided by factory f for job i ; 0 otherwise
E_i^f	The transportation time of job i from factory f to its destination
Variables	
D_i^f	1 if job i is delivered to factory f ; 0 otherwise
R_{ir}^f	1 if the l th link of the r th OR-node of job i is selected; 0 otherwise
X_{ij}^f	1 if O_{ij}^f is selected; 0 otherwise
Z_{ijk}^f	1 if O_{ij}^f is processed on machine k ; 0 otherwise
$Y_{ijj'}^f$	1 if O_{ij}^f is processed before $O_{ij'}^f$; 0 otherwise
S_{ij}^f	The starting time of O_{ij}^f
C_i	The completion time of job i
C_{\max}	Makespan
MMW	The maximal machine workload
TWM	The total workload of the machines

Objectives:

(1) Minimize the maximum completion time:

$$\min f_1 = C_{\max} \tag{1}$$

(2) Minimize the maximal machine workload:

$$\min f_2 = MMW \tag{2}$$

(3) Minimize the total workload of the machines:

$$\min f_3 = TWM \tag{3}$$

Constraints:

Each job can only be assigned to one factory:

$$\sum_f D_i^f = 1, \quad \forall i \tag{4}$$

Only one link of an OR-node can be selected. If the job is not assigned to factory f , the network with the OR-nodes provided by factory f for job i will not be considered:

$$\sum_l R_{ir}^f = D_i^f, \quad \forall f, i, r \tag{5}$$

$$\sum_j X_{ij}^f = D_i^f, \quad \forall f, i \tag{6}$$

If the link where operation O_{ij}^f is located is not traversed, O_{ij}^f will not be selected.

$$X_{ij}^f \leq M \cdot (1 - W_{ijrl}^f) + M \cdot R_{ir}^f, \quad \forall f, i, j, r, l \tag{7}$$

After analyzing the premises of selecting the operations, it can be concluded that if an operation is not controlled by any OR-node, it will certainly be selected, and it will also be selected if all its controlling links are traversed. Therefore, this paper formulates the OR-node function constraint as follows:

$$X_{ij}^f \geq 1 - M \cdot \sum_r \sum_l W_{ijrl}^f \cdot (1 - R_{ir}^f), \quad \forall f, i, j \tag{8}$$

O_{ij}^f will be assigned to one machine selected from its alternative machine set.

$$\sum_k Z_{ijk}^f = X_{ij}^f, \quad \forall f, i, j \tag{9}$$

The precedence relationship 0–1 variable $Y_{ijj'}^f$ should be constrained according to the precedence constraints expressed by the networks.

$$\sum_{i'} \sum_j \sum_{j'} Y_{ijj'}^f \leq M \cdot D_i^f, \quad \forall f, i \tag{10}$$

$$\sum_i \sum_j \sum_{j'} Y_{ijj'}^f \leq M \cdot D_{i'}^f, \quad \forall f, i' \tag{11}$$

$$M \cdot (2 - X_{ij}^f - X_{ij'}^f) + Y_{ijj'}^f \geq U_{ijj'}^f, \quad \forall f, i, j, j' \tag{12}$$

The operations belonging to the same job should be processed sequentially according to the precedence relationships.

$$S_{ij'}^f \geq S_{ij}^f + \sum_k P_{ijk}^f \cdot Z_{ijk}^f - M \cdot (3 - X_{ij}^f - X_{ij'}^f - Y_{ijj'}^f), \quad \forall f, i, j, j' \tag{13}$$

$$S_{ij}^f \geq S_{ij'}^f + \sum_k P_{ij'k}^f \cdot Z_{ij'k}^f - M \cdot (2 - X_{ij}^f - X_{ij'}^f + Y_{ijij'}^f), \quad \forall f, i, j, j' \tag{14}$$

The operations assigned to the same machine should also obey the precedence relationships.

$$S_{ij'}^f \geq S_{ij}^f + P_{ijk}^f - M \cdot (3 - Z_{ijk}^f - Z_{i'j'k'}^f - Y_{ijij'}^f), \quad \forall f, i, i', j, j', k \tag{15}$$

$$S_{i'j'}^f \geq S_{i'j}^f + P_{i'j'k}^f - M \cdot (2 - Z_{i'jk}^f - Z_{i'j'k'}^f + Y_{ijij'}^f), \quad \forall f, i, i', j, j', k \tag{16}$$

If job i is not processed in factory f , S_{ij}^f will not be considered and is set to 0:

$$\sum_j S_{ij}^f \leq M \cdot D_i^f, \quad \forall f, i \tag{17}$$

From the above, the completion time can be obtained as:

$$C_{\max} \geq S_{ij}^f + \sum_k P_{ijk}^f \cdot Z_{ijk}^f - M \cdot (1 - X_{ij}^f) + E_i^f \cdot D_i^f, \quad \forall f, i, j \tag{18}$$

The *MMW* value can be obtained as:

$$MMW = \max_{\forall f,k} \left(\sum_i \sum_j P_{ijk}^f \cdot Z_{ijk}^f \right) \tag{19}$$

The *TWM* value can be obtained as:

$$TWM = \sum_f \sum_k \sum_i \sum_j P_{ijk}^f \cdot Z_{ijk}^f \tag{20}$$

4 The proposed MOMA

4.1 Multiobjective memetic algorithm

In contrast to the GA, which simulates biological evolutionary procedures, Moscato et al. proposed the MA [4], which simulates cultural evolutionary procedures. The MA uses local heuristic search to simulate mutation and is supported by a large amount of expertise, so the MA can be considered a combination of population-based global search and individual-based local heuristic search (Table 2). This combination mechanism of the MA makes its search efficiency

greater than that of the traditional GA in some optimization problems, and it can be applied to a wide range of problem areas with satisfactory results. The crossover operator and mutation operators for individuals are designed according to the encoding method. In addition, the MOMA deals with multiple objectives in a Pareto-based way to find the optimal Pareto frontier. The Pareto frontier and Pareto dominating relationships have been widely introduced in multiobjective optimization publications and will not be repeated in this paper. The flow chart of the MOMA is given in Fig. 2.

The concrete steps of the proposed MOMA are shown below:

Step 1: Initialization. Set the parameters and generate the population randomly. The generation number is $Gen = 1$.

Step 2: Select individuals through tournament selection, and perform crossover with probability P_C and mutation with probability P_M .

Step 3: For each individual X_{cur} , perform an SA search.

Step 4: Evaluate the population and generate the Pareto frontier. Determine whether the termination criteria are satisfied; if so, go to step 5, and otherwise, go to step 3.

Step 5: Output the solutions of the current Pareto frontier. End.

4.2 Coding methods

The encoding method proposed in this paper includes four types of chromosomes: factory chromosomes, OR-node chromosomes, operation chromosomes, and machine chromosomes (Fig. 3). Except for the factory chromosome, all types of chromosomes have subchromosomes corresponding to factories, as shown in Fig. 4. The factory chromosome is used to represent the factory selected for the job. The OR-node chromosome represents the selection state of the OR-nodes in the network. ‘1’ and ‘2’ represent the left and right links, respectively. For the operation chromosome, each gene contains both a job number and an operation number. The decoding order is from left to right. The machine chromosome, arranged in the order of the job numbers, represents the machine assigned by the operation at each position. In conclusion, the complete production scheme is decoded from the above example: factory 1 is responsible for processing jobs 1 and 3, while factory 2 is responsible for job 2. The production order of the operations in factory 1 is 1–1(1) → 3–2(2) → 1–2(4) → 1–3(2) → 3–3(1) → 3–7(5) → 3–5(4) → 3–6(4), and that in factory 2 is 2–3(2) → 2–1(1) → 2–4(4) → 2–6(3). The pair of numbers outside the parentheses represents a job and its operation number, and the number inside the parentheses is the assigned machine number. According to the decoding method described in the previous publication [16], the objective makespan can be obtained easily.

Table 2 The MODIPPS example problem

Job	Factory 1		Factory 2	
	Operation	Alternative machines	Operation	Alternative machines
1	1	<u>1</u> , 2	1	1, 3
	2	2, <u>4</u>	2	3, 4
	3	<u>2</u> , 3	3	2, 4
2	1	3, 5	1	<u>2</u> , 4
	2	3, 4	2	3, <u>4</u>
	3	1, 5	3	<u>2</u> , 5
	4	1, 4, 5	4	1, 2, <u>4</u>
3	1	<u>2</u> , 3	1	2, 3
	2	<u>2</u> , 4	2	3, 4
	3	<u>1</u> , 5	3	2, 5
	4	1, <u>3</u>	4	1, 2
	5	1, 3, <u>4</u>	5	2, 3, 4
	6	2, <u>4</u>	6	2, 3

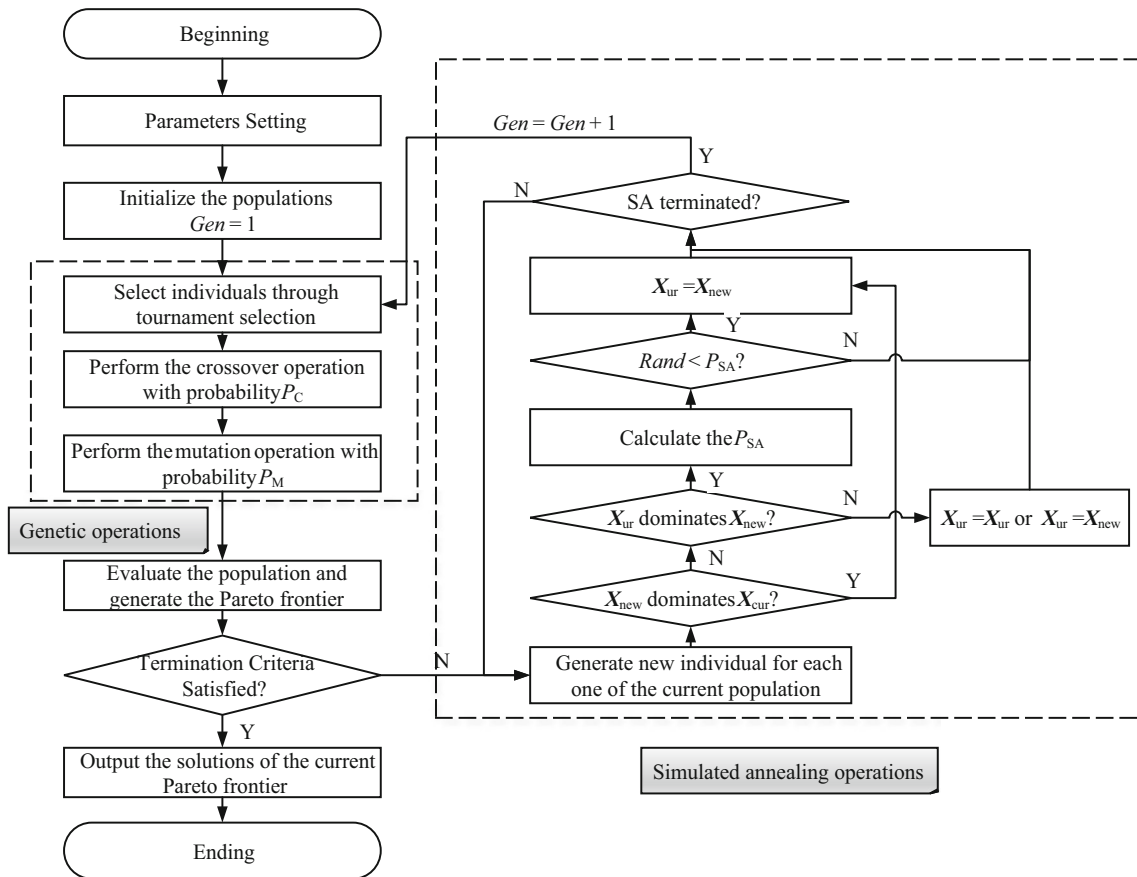


Fig. 2 The flow chart of the proposed MOMA

4.3 Initialization

Each individual in the population is randomly initialized according to the above encoding rules, and each individual represents a solution. All individuals should obey the

precedence constraints to ensure the smooth operation of the MOMA. Without infeasible individuals in the population, in other words, there is no need to adjust the operation chromosome of the infeasible individuals after each iteration. This

Fig. 3 Example process network graphs

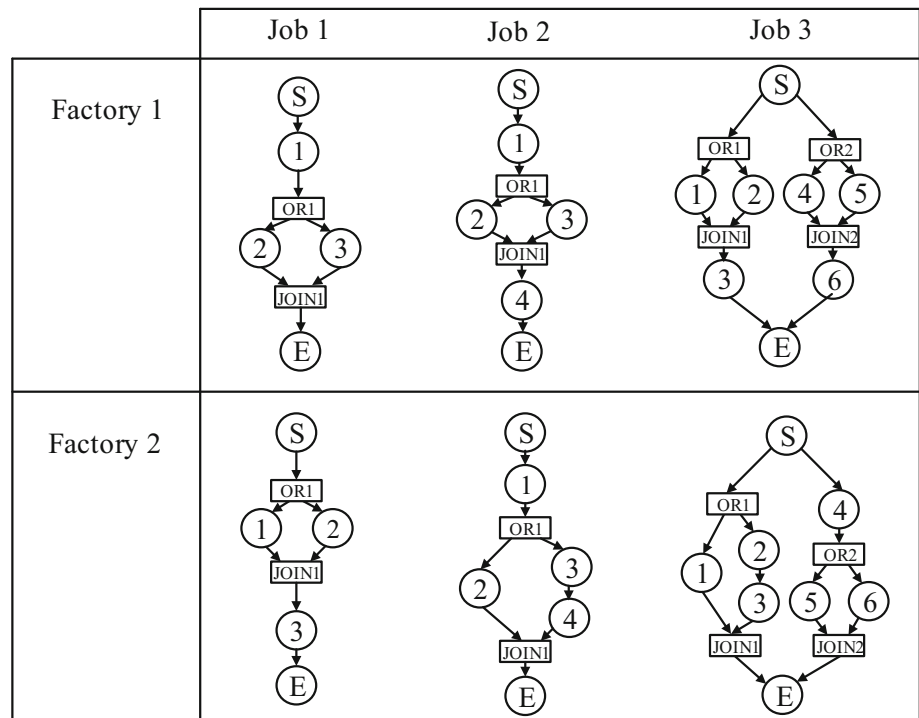
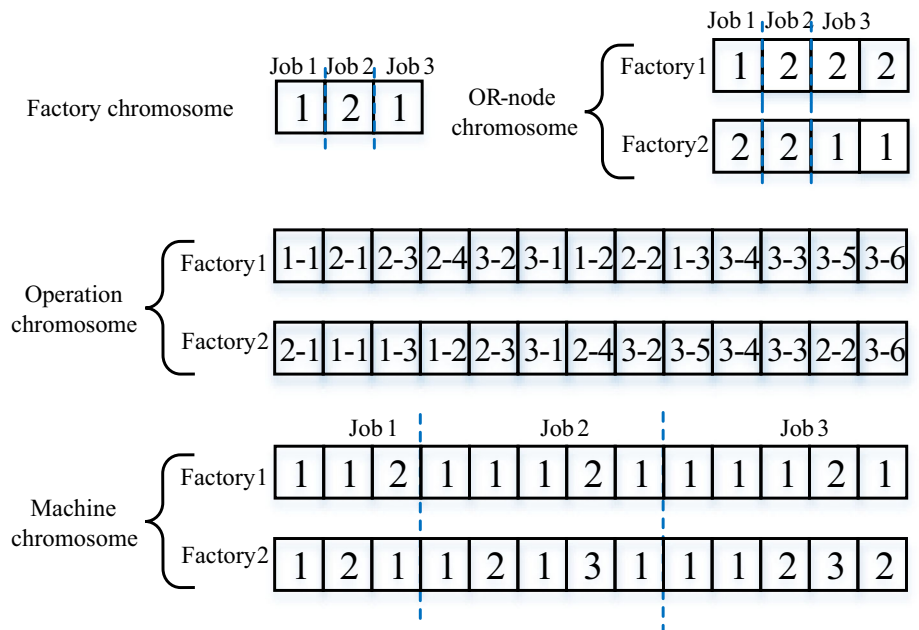


Fig. 4 The coding method for a MOIPPS scheme



saves adjustment time for the algorithm to a certain extent and can accelerate the solving speed of the algorithm.

4.4 Crossover

A new individual is obtained by the crossover operator between two parent individuals, which are selected by the tournament selection method. The crossover method is shown in Fig. 5. Two points are randomly selected, and the

genes outside the points are transferred to the new individual at the same positions, while the sequence of genes inside the points is obtained by mapping. This mapping operation can generate an offspring chromosome with the same legality from two feasible operation chromosomes to ensure continuous legitimacy in the evolutionary process of the MOMA. For factory, OR-node, and machine chromosomes that are not constrained by precedence, their offspring chromosomes can be obtained by swapping the subsequences in the corre-

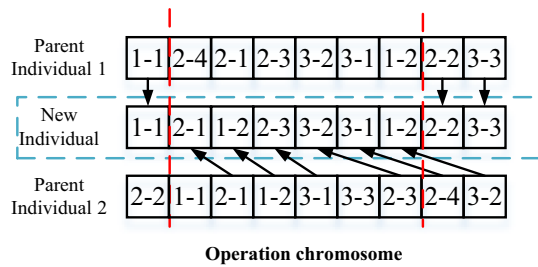


Fig. 5 Crossover for the operation chromosome

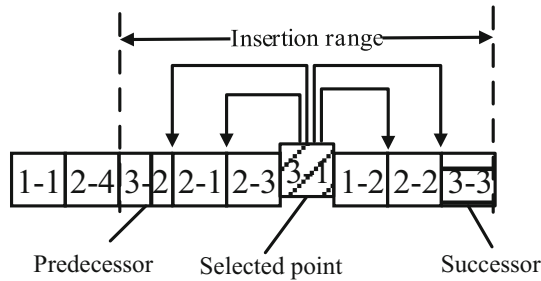


Fig. 6 Mutation for operation chromosome

sponding positions. This operation is relatively simple, so it will not be described here.

4.5 Mutation

This paper designs a single-point insertion operation as the mutation operator, as shown in Fig. 6. A point is randomly selected to determine the range where it can be inserted, i.e., its predecessor operation and successor operation. The selected operation point can be inserted at any position in the range. The factory chromosome, OR-node chromosome, and machine chromosome are not constrained by the precedence constraints, so the mutation can be performed through single-point switching to randomly select a point and switch the selection state to one of the other states. The switching operation is shown in Fig. 7.

4.6 Simulated annealing mechanism

The proposed algorithm has a powerful global search capability, but it easily falls into a local optimum, so the SA mechanism for local search is introduced here to address the shortcomings of the algorithm. In the SA step, new individuals are generated by the above two genetic operations. The concrete steps of SA are shown below.

For each individual X_{cur} , perform SA search:

Step 1: Randomly select an individual to carry out the crossover operation with the current individual X_{cur} , and then carry out mutation on the current individual to generate a new individual X_{new} .

Step 2: Determine whether X_{new} dominates X_{cur} . If so, set $X_{cur} = X_{new}$ and go to step 5; otherwise, go to step 3.

Step 3: Determine whether X_{cur} dominates X_{new} . If so, go to step 4; otherwise, set $X_{cur} = X_{new}$ or $X_{new} = X_{cur}$ and go to step 5.

Step 4: Calculate P_{SA} and generate a $[0, 1]$ random value $Rand$. Determine whether $Rand < P_{SA}$. If so, set $X_{cur} = X_{new}$; otherwise, go to step 5.

Step 5: Determine whether SA has terminated. If so, go to step 6; otherwise, go to step 1.

Step 6: Output the current solution X_{cur} . End.

The P_{SA} of the single-objective optimization problem is calculated as follows, where f^{new} and f^{cur} denote the fitness values of the new individual and the current individual:

$$P_{SA} = \begin{cases} 1, & f^{new} < f^{cur} \\ \exp\left(-\frac{f^{new} - f^{cur}}{T}\right), & f^{new} \geq f^{cur} \end{cases} \quad (21)$$

Since this paper studies multiobjective optimization, the above P_{SA} formula for single-objective problems cannot be applied directly to this multiobjective problem. The P_{SA} formula for the multiobjective problem is shown below, and the subscript m denotes the m th optimization objective in the multiobjective optimization problem.

$$P_{SA} = \begin{cases} 1, & X_{new} \text{ dominates } X_{cur} \\ \prod_{m=1}^M \exp\left(-\frac{f_m}{T}\right), & otherwise \end{cases} \quad (22)$$

The value T in Eqs. (21) and (22), which is the current temperature, and the intermediate variable f_m can be calculated as follows:

$$f_m = \begin{cases} 0, & \text{if } f_m^{new} < f_m^{cur} \\ f_m^{new} - f_m^{cur}, & otherwise \end{cases} \quad (23)$$

$$T = \alpha^k \cdot T_0 \quad (24)$$

where α is the cooling rate; k is the number of cooling times, which is set to the current iteration number of the MOMA; and T_0 is the initial temperature.

5 Experimental studies and discussion

To test the proposed MOMA, two groups of numerical experiments are designed. The data of these numerical experiments are openly available in existing publications, so the experimental results can objectively illustrate the superiority of the proposed algorithm. The cases are divided into two groups according to the size of the experimental case: the first group is from the literature [35, 36], and the second is adopted

Fig. 7 Mutation for the three chromosomes

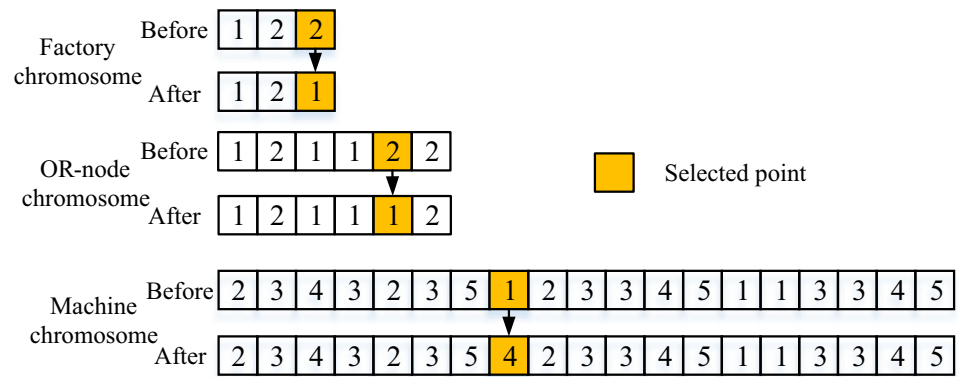


Table 3 Orthogonal array and average values

No	Parameters					Avg. of IGD
	PopSize	P_m	P_c	α	T_0	
1	100	0.2	0.2	0.99	500	46.7
2	100	0.4	0.4	0.98	400	60.8
3	100	0.6	0.6	0.97	300	69.2
4	100	0.8	0.8	0.96	200	77.7
5	200	0.2	0.4	0.97	200	35.3
6	200	0.4	0.2	0.96	300	47.6
7	200	0.6	0.8	0.99	400	59.1
8	200	0.8	0.6	0.98	500	63.6
9	300	0.2	0.6	0.96	400	20.9
10	300	0.4	0.8	0.97	500	22.2
11	300	0.6	0.2	0.98	200	39.8
12	300	0.8	0.4	0.99	300	40.3
13	400	0.2	0.8	0.98	300	20.1
14	400	0.4	0.6	0.99	200	11.1
15	400	0.6	0.4	0.96	500	15.6
16	400	0.8	0.2	0.97	400	10.7

from [13]. The case data can be downloaded from the relevant website to facilitate comparative studies by future researchers. The proposed MOMA is implemented via C++ programming and is run on a PC with an i7-8700 CPU and 16 GB memory. The classic NSGA-II and MOEA/D algorithms are employed for comparison with the proposed MOMA. In addition, a version of the MOMA without an SA mechanism, named MOMA_NSA, is employed to prove the effectiveness of SA in the MOMA. Each algorithm is run 10 times independently with a fixed running time of 300 s. The three performance indicators for the multiobjective algorithms are the set coverage (CS), generational distance (GD), and inverse generational distance (IGD). CS was proposed by Zitzler et al. [41] to compare the coverage rates of two solution sets. A larger CS value means that there are more solutions in this solution set that dominate those in the other set, which also means that this set is closer to the optimal Pareto frontier. The second evaluation indicator is GD [30],

which is used to describe the distance between the current Pareto frontier and the optimal Pareto frontier. The IGD indicator is a comprehensive evaluation index that can reflect the convergence and distribution degrees [2]. The smaller the IGD value is, the better the comprehensive performance of the MOEA is.

(1) CS. Suppose both sets A and B are approximations of the Pareto frontier. $C(A, B) > C(B, A)$ means that set A is a better approximation of the optimal Pareto frontier. Then, $C(A, B)$ is defined by Eq. (25).

$$C(A, B) = \frac{|\{b \in B | \exists a \in A : a \text{ dominates } b\}|}{|B|} \tag{25}$$

(2) GD. By calculating the distance between a point on the current Pareto frontier and the optimal Pareto frontier, the solution set obtained by the algorithm is evaluated. The

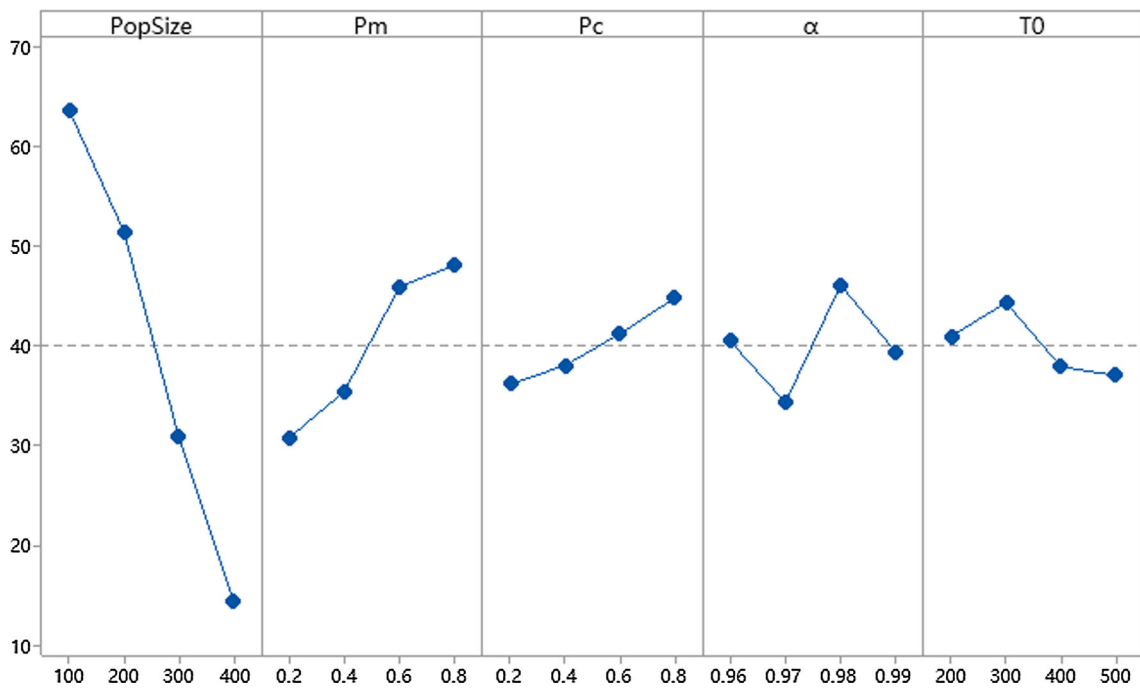


Fig. 8 Main effects plot for MOMA

Table 4 The comparison based on GD

Problem	No. of factories/jobs/machines	GD	MOEA/D	NSGA-II	MOMA_NSA	MOMA
P1	2/4/18	Average	84.76	3.59	66.37	0.88
		Std	14.81	4.04	12.92	1.63
P2	2/6/10	Average	96.14	3.52	66.19	3.33
		Std	40.65	5.63	13.40	3.52

The results in bold are better

Table 5 The comparison based on IGD

Problem	No. of factories/jobs/machines	IGD	MOEA/D	NSGA-II	MOMA_NSA	MOMA
P1	2/4/18	Average	79.02	6.21	77.02	1.81
		Std	12.17	2.82	12.91	1.79
P2	2/6/10	Average	83.81	10.03	74.66	4.82
		Std	20.67	7.80	20.17	4.52

The results in bold are better

smaller the GD value is, the better the current Pareto frontier. GD is defined as follows:

$$GD = \frac{\sqrt{\sum_{i=1}^n d_i^2}}{n}, \tag{26}$$

where d_i is the Euclidean distance from the i th point of the current Pareto frontier (PF) to the nearest point of the optimal Pareto frontier (PF^*) and n is the number of solutions in PF . Since the optimal frontier cannot be obtained directly, PF^* here is the optimal Pareto frontier composed of all the solutions of various multiobjective evolutionary algorithms after each independent run.

(3) IGD. The shortest distance from a point on PF^* to PF is calculated to evaluate the algorithms. IGD is defined as follows:

$$IGD = \frac{\sqrt{\sum_{i \in PF^*} d_i^{*2}}}{n}, \tag{27}$$

where d_i^* is the Euclidean distance from the i th point of PF^* to the nearest point of PF and n is the number of solutions in PF^* . Since the optimal frontier cannot be obtained directly, PF^* here is the optimal Pareto frontier composed of all the solutions of various multiobjective evolutionary algorithms.

Table 6 The comparison based on CS (part 1)

Problem	No. of factories/jobs/machines	CS	A:MOEA/D B:NSGA-II		A:MOEA/D B:MOMA		A:NSGA-II B:MOMA	
			C(A, B)	C(B, A)	C(A, B)	C(B, A)	C(A, B)	C(B, A)
			P1	2/4/18	Average	0	1.00	0
		Std	0	0	0	0	0.28	0.48
P2	2/6/10	Average	0	1.00	0	1.00	0.31	0.31
		Std	0	0	0	0	0.32	0.48

The results in bold are better

Table 7 The comparison based on CS (part 2)

Problem	No. of factories/jobs/machines	CS	A:MOEA/D B: MOMA_NSA		A: NSGA-II B: MOMA_NSA		A: MOMA_NSA B:MOMA	
			C(A, B)	C(B, A)	C(A, B)	C(B, A)	C(A, B)	C(B, A)
			P1	2/4/18	Average	0.27	0.46	1.00
		Std	0.21	0.32	0	0	0	0
P2	2/6/10	Average	0	1.00	0	1.00	0.31	0.31
		Std	0	0	0	0	0.32	0.48

The results in bold are better

Table 8 The comparison based on GD

Problem	No. of factories/jobs/machines	GD	MOEA/D	NSGA-II	MOMA_NSA	MOMA
E1	2/24/18	Average	545.30	295.94	609.8	0.00
		Std	155.06	54.38	161.0	0.00
E2	2/36/10	Average	716.38	409.83	678.68	0.00
		Std	203.54	172.59	141.67	0.00
E3	2/60/9	Average	2214.25	1588.92	2531.29	24.38
		Std	562.37	1053.97	617.66	77.10
E4	2/30/3	Average	1058.40	211.52	997.01	0.00
		Std	155.93	90.60	194.38	0.00
E5	2/60/6	Average	49.05	35.09	52.52	0.00
		Std	10.17	13.84	6.53	0.00
E6	2/48/5	Average	31.05	1.88	35.69	0.15
		Std	3.15	2.30	7.20	0.47
E7	2/30/5	Average	24.31	5.05	24.69	0.00
		Std	5.34	3.19	4.30	0.00
E8	2/36/5	Average	10.87	6.41	12.00	0.00
		Std	1.95	2.25	2.25	0.00
E9	2/30/6	Average	39.67	12.59	36.77	0.00
		Std	9.02	11.06	8.93	0.00
E10	2/30/5	Average	6.10	1.39	5.75	0.00
		Std	0.54	1.58	0.97	0.00
E11	2/48/6	Average	23.42	4.68	21.80	0.00
		Std	2.90	2.21	4.28	0.00

The results in bold are better

Table 9 The comparison based on IGD

Problem	No. of factories/jobs/machines	IGD	MOEA/D	NSGA-II	MOMA_NSA	MOMA
E1	2/24/18	Average	409.44	218.80	399.67	0
		Std	149.35	73.92	148.54	0
E2	2/36/10	Average	404.42	245.23	399.15	0
		Std	173.30	116.64	160.31	0
E3	2/60/9	Average	1412.33	1292.52	1445.41	27.05
		Std	644.41	756.86	825.45	62.57
E4	2/30/3	Average	1033.26	227.02	1016.33	0
		Std	244.61	88.67	253.10	0
E5	2/60/6	Average	18.55	14.61	18.05	0
		Std	3.23	3.86	3.32	0
E6	2/48/5	Average	28.44	7.63	26.55	1.48
		Std	8.50	3.95	7.49	1.60
E7	2/30/5	Average	24.93	10.72	25.83	0.53
		Std	5.36	5.02	7.15	1.05
E8	2/36/5	Average	13.96	13.32	14.36	0
		Std	1.42	4.49	1.52	0
E9	2/30/6	Average	37.61	19.90	39.12	1.14
		Std	7.76	5.20	7.81	2.68
E10	2/30/5	Average	7.32	6.29	6.43	1.21
		Std	2.52	3.44	1.48	0.79
E11	2/48/6	Average	24.99	12.78	26.11	0.58
		Std	5.17	5.24	4.79	1.16

The results in bold are better

The parameter configuration affects the performance of the MOMA. The parameters are the population size *PopSize*, the crossover probability P_c , the mutation probability P_m , the cooling rate α , and the initial temperature T_0 . A Taguchi design-of-experiment (DOE) approach is adopted to determine the most appropriate parameter configuration. The level of each parameter is as follows: $PopSize = \{100, 200, 300, 400\}$, $P_m = \{0.2, 0.4, 0.6, 0.8\}$, $P_c = \{0.2, 0.4, 0.6, 0.8\}$, $\alpha = \{0.96, 0.97, 0.98, 0.99\}$, and $T_0 = \{200, 300, 400, 500\}$. An orthogonal array $L16(4^5)$ is used, as shown in Table 3. The comprehensive index IGD is selected to evaluate the parameter combinations. According to Fig. 8, the MOMA performs better with the parameter combination $PopSize = 400$, $P_m = 0.2$, $P_c = 0.2$, $\alpha = 0.97$, and $T_0 = 500$.

5.1 Experiment 1

Cases P1 and P2 in this group of experiments are small-scale DIPPS problems, which are derived from the literature [35, 36]. By comparing the indicators GD, IGD, and CS, as shown in Tables 4, 5, the proposed MOMA can obtain a better Pareto frontier with smaller GD and IGD values than those of the other comparison algorithms, indicating that the frontier of the MOMA is closer to the optimal one. According to the value of the indicator CS, the number of dominating solu-

tions in the MOMA solution set is greater than the number of dominated solutions when compared with the other two algorithms. Therefore, the proposed MOMA performed better than NSGA-II, MOMA_NSA and MOEA/D.

5.2 Experiment 2

This group of experiments contains 11 cases. The numbers of factories, jobs, and machines in these examples are shown in the Tables 6, 7, and the cases that belong to the large-scale DIPPS problem are indicated. The comparison results for NSGA-II, MOMA_NSA and MOEA/D are shown in Tables 8, 9, 10, 11. The MOMA has smaller GD and IGD values and is superior to the comparison algorithms in terms of the indicator CS. Taking case E11 as an example, its Gantt chart and the Pareto front graphs of the three algorithms are as shown in Figs. 9 and 10.

5.3 Discussions

From the above experimental results and the three comparative indicators, it can be concluded that the MOMA proposed in this paper outperforms the comparative algorithms in the different testing instances. Compared with MOEA/D and NSGA2, the MOMA has not only a strong global search abil-

Table 10 The comparison based on coverage (part 1)

Problem	No. of factories/jobs/machines	Coverage	A:MOEA/D B:NSGA-II		A:MOEA/D B:MOMA		A:NSGA-II B:MOMA	
			C(A, B)	C(B, A)	C(A, B)	C(B, A)	C(A, B)	C(B, A)
			E1	2/24/18	Average	0	1.00	0
		Std	0	0	0	0	0	0
E2	2/36/10	Average	0	1.00	0	1.00	0	1.00
		Std	0	0	0	0	0	0
E3	2/60/9	Average	0	0.82	0	0.99	0.06	0.8
		Std	0	0.20	0	0.04	0.20	0.42
E4	2/30/3	Average	0	1.00	0	1.00	0	1.00
		Std	0	0	0	0	0	0
E5	2/60/6	Average	0	0.69	0	1.00	0	1.00
		Std	0	0.26	0	0	0	0
E6	2/48/5	Average	0	1.00	0	1.00	0.03	0.46
		Std	0	0	0	0	0.08	0.49
E7	2/30/5	Average	0	1.00	0	1.00	0	0.90
		Std	0	0	0	0	0	0.16
E8	2/36/5	Average	0	0.97	0	1.00	0	1.00
		Std	0	0.09	0	0	0	0
E9	2/30/6	Average	0	1.00	0	1.00	0	0.81
		Std	0	0	0	0	0	0.36
E10	2/30/5	Average	0	0.89	0	0.98	0	0.53
		Std	0	0.17	0	0.06	0	0.43
E11	2/48/6	Average	0	1.00	0	1.00	0	0.83
		Std	0	0	0	0	0	0.33

The results in bold are better

Table 11 The comparison based on coverage (part 2)

Problem	No. of factories/jobs/machines	Coverage	A:MOEA/D B: MOMA_NSA		A: NSGA-II B: MOMA_NSA		A: MOMA_NSA B:MOMA	
			C(A, B)	C(B, A)	C(A, B)	C(B, A)	C(A, B)	C(B, A)
			E1	2/24/18	Average	0.14	0.48	1.00
		Std	0.23	0.29	0	0	0	0
E2	2/36/10	Average	0.40	0.38	1.00	0	0	1.00
		Std	0.38	0.42	0	0	0	0
E3	2/60/9	Average	0.48	0.36	0.80	0	0	1.00
		Std	0.22	0.32	0.27	0	0	0
E4	2/30/3	Average	0.40	0.28	1.00	0	0	1.00
		Std	0.36	0.34	0	0	0	0
E5	2/60/6	Average	0.27	0.33	0.69	0	0	1.00
		Std	0.26	0.20	0.26	0	0	0
E6	2/48/5	Average	0.24	0.53	1.00	0	0	1.00
		Std	0.22	0.33	0	0	0	0

Table 11 continued

Problem	No. of factories/jobs/machines	Coverage	A:MOEA/D B: MOMA_NSA		A: NSGA-II B: MOMA_NSA		A: MOMA_NSA B:MOMA	
			C(A, B)	C(B, A)	C(A, B)	C(B, A)	C(A, B)	C(B, A)
			E7	2/30/5	Average	0.31	0.43	1.00
		Std	0.31	0.30	0	0	0	0
E8	2/36/5	Average	0.33	0.33	0.96	0	0	1.00
		Std	0.29	0.24	0.12	0	0	0
E9	2/30/6	Average	0.46	0.26	1.00	0	0	1.00
		Std	0.31	0.23	0	0	0	0
E10	2/30/5	Average	0.35	0.57	0.87	0	0	0.97
		Std	0.23	0.20	0.20	0	0	0.07
E11	2/48/6	Average	0.50	0.34	1.00	0	0	1.00
		Std	0.34	0.32	0	0	0	0

The results in bold are better

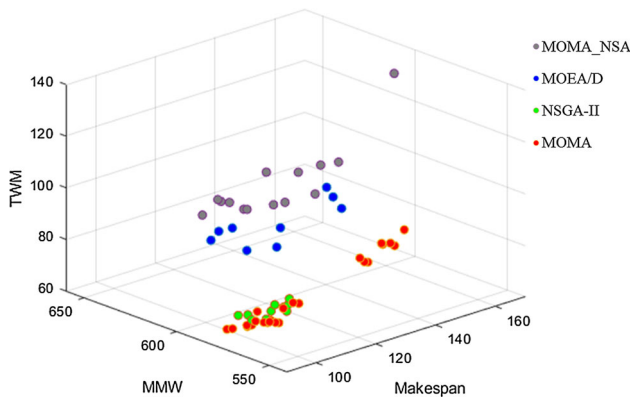


Fig. 9 Pareto frontier graph of three algorithms of case E11

ity but also a better local search ability. By introducing the SA mechanism for local search in a specific neighborhood, the MOMA should accept relatively poor solutions with a certain probability so that it converges at a certain speed and ensures the diversity of the population. Therefore, the algorithm can avoid falling into a local optimum prematurely and can guarantee the solution quality. The balance of exploration and exploitation abilities is the MOMA’s important characteristic, and it is also the key to obtaining better solutions.

6 Conclusions

This paper describes a new manufacturing mode in a distributed manufacturing system, which integrates the process planning subsystem with the scheduling subsystem. Owing to the complementarity of the two subsystems, the production efficiency and the quality can be significantly improved. This method could provide more potential choices for production managers to save costs and keep production steady. Furthermore, it is difficult for operators to make decisions on multiobjective problems in actual production, and actual problems are always multiobjective problems. For the MODIPPS problem, a novel network-based MILP model is established for the first time, and the MOMA is designed by taking advantage of the network and the OR-node logic. The encoding and decoding method is newly designed for the distributed manufacturing mode to integrate the process planning and scheduling steps. Owing to the introduction of the SA mechanism to enhance the exploration and exploitation ability, the proposed MOMA outperforms the other two classic multiobjective algorithms.

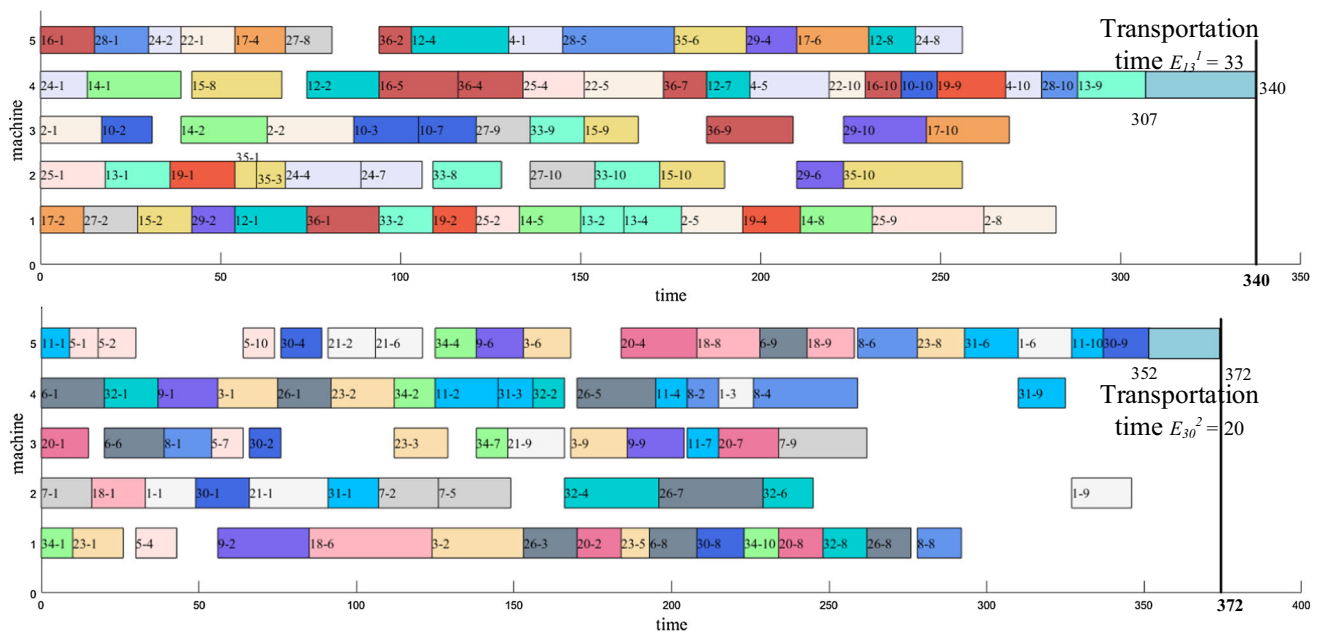


Fig. 10 The Gantt chart for case E11

The MOMA proposed in this paper can provide managers with a near-optimal Pareto frontier and greatly reduces the difficulty of decision-making for multiobjective production problems. The integration mode can accelerate the intelligentization of modern manufacturing systems and can greatly improve the production management capacity of enterprises. This successful application should inspire managers to consider and deal with production problems from a global perspective and explore their solutions from the perspective of the essential problem characteristics.

Acknowledgements This work is supported in part by the National Key R&D Program of China under Grant No. 2019YFB1704600, in part by the National Natural Science Foundation of China under Grant Nos. 51825502 and U21B2029, and in part by the Program for HUST Academic Frontier Youth Team under Grant 2017QYTD04.

Declarations

Conflict of interest Qihao Liu, Xinyu Li, Liang Gao declare that they have no conflict of interest or financial conflicts to disclose.

References

- Barzanji R, Naderi B, Begen MA (2019) Decomposition algorithms for the integrated process planning and scheduling problem. *Omega* 93:102025
- Czyżżak P, Jaskiewicz A (1998) Pareto simulated annealing—a metaheuristic technique for multiple-objective combinatorial optimization. *J Multi-Criteria Decis Anal* 7(1):34–47
- De Giovanni L, Pezzella F (2010) An improved genetic algorithm for the distributed and flexible job-shop scheduling problem. *Eur J Oper Res* 200(2):395–408
- Franca PM, Mendes A, Moscato P (2001) A memetic algorithm for the total tardiness single machine scheduling problem. *Eur J Oper Res* 132(1):224–242
- Jia HZ, Fuh JYH, Nee AYC, Zhang YF (2002) Web-based multi-functional scheduling system for a distributed manufacturing environment. *Concurr Eng Res Appl* 10(1):27–39
- Jin L, Zhang C, Shao X, Yang X, Tian G (2016) A multi-objective memetic algorithm for integrated process planning and scheduling. *Int J Adv Manuf Technol* 85(5–8):1513–1528
- Li H, Xinyu L, Liang G (2021) A discrete artificial bee colony algorithm for the distributed heterogeneous no-wait flowshop scheduling problem. *Appl Soft Comput* 100:106946
- Li X, Gao L, Li W (2012) Application of game theory based hybrid algorithm for multi-objective integrated process planning and scheduling. *Expert Syst Appl* 39(1):288–297
- Li X, Xiao S, Wang C, Yi J (2019) Mathematical modeling and a discrete artificial bee colony algorithm for the welding shop scheduling problem. *Memet Comput* 11(4):371–389
- Li Y, Ba L, Cao Y, Liu Y, Yang M (2015) Research on integrated process planning and scheduling problem with consideration of multi-objectives. *China Mech Eng* 26(17):2344
- Li Y, Li X, Gao L, Meng L (2020) An improved artificial bee colony algorithm for distributed heterogeneous hybrid flowshop scheduling problem with sequence-dependent setup times. *Comput Ind Eng* 147:106638
- Liang J, Wang P, Guo L, Qu B, Yue C, Yu K, Wang Y (2019) Multi-objective flow shop scheduling with limited buffers using hybrid self-adaptive differential evolution. *Memet Comput* 11(4):407–422
- Lin C-S, Li P-Y, Wei J-M, Wu M-C (2020) Integration of process planning and scheduling for distributed flexible job shops. *Comput Oper Res* 124:105053
- Liu Q, Li X, Gao L (2021) Mathematical modeling and a hybrid evolutionary algorithm for process planning. *J Intell Manuf* 32(3):781–797

15. Liu Q, Li X, Gao L (2021) A novel MILP model based on the topology of a network graph for process planning in an intelligent manufacturing system. *Engineering* 7(6):807–817
16. Liu Q, Li X, Gao L, Li Y (2020) A modified genetic algorithm with new encoding and decoding methods for integrated process planning and scheduling problem. *IEEE Trans Cybern* 51(9):4429–4438
17. Liu XJ, Yi H, Ni ZH (2013) Application of ant colony optimization algorithm in process planning optimization. *J Intell Manuf* 24(1):1–13
18. Lu C, Gao L, Gong W, Hu C, Li X (2021) Sustainable scheduling of distributed permutation flow-shop with non-identical factory using a knowledge-based multi-objective memetic optimization algorithm. *Swarm Evol Comput* 60:100803
19. Meng L, Zhang C, Ren Y, Zhang B, Lv C (2020) Mixed-integer linear programming and constraint programming formulations for solving distributed flexible job shop scheduling problem. *Comput Ind Eng* 142:106347
20. Mohammadi G, Karampourhaghghi A, Samaei F (2012) A multi-objective optimisation model to integrating flexible process planning and scheduling based on hybrid multi-objective simulated annealing. *Int J Prod Res* 50(18):5063–5076
21. Mohapatra P, Benyoucef L, Tiwari MK (2013) Integration of process planning and scheduling through adaptive setup planning: a multi-objective approach. *Int J Prod Res* 51(23–24):7190–7208
22. Mohapatra P, Nayak A, Kumar SK, Tiwari MK (2015) Multi-objective process planning and scheduling using controlled elitist non-dominated sorting genetic algorithm. *Int J Prod Res* 53(6):1712–1735
23. Na De Ri B, Ruiz R (2010) The distributed permutation flowshop scheduling problem. *Comput Oper Res* 37(4):754–768
24. Okwudire CE, Madhyastha HV (2021) Distributed manufacturing for and by the masses. *Science (New York, N. Y.)* 372(6540):341–342
25. Phu-ang A, Thammano A (2017) Memetic algorithm based on marriage in honey bees optimization for flexible job shop scheduling problem. *Memet Comput* 9(4):295–309
26. Qin H, Li T, Teng Y, Wang K (2021) Integrated production and distribution scheduling in distributed hybrid flow shops. *Memet Comput* 13(2):185–202
27. Raeesi NMR, Kobti Z (2012) A memetic algorithm for job shop scheduling using a critical-path-based local search heuristic. *Memet Comput* 4(3):231–245
28. Shen W, Wang L, Hao Q (2006) Agent-based distributed manufacturing process planning and scheduling: a state-of-the-art survey. *IEEE Trans Syst Man Cybern Part C Appl Rev* 36(4):563–577
29. Shokouhi E (2018) Integrated multi-objective process planning and flexible job shop scheduling considering precedence constraints. *Prod Manuf Res Open Access J* 6(1):61–89
30. Van Veldhuizen DA, Lamont GB (1998) Evolutionary computation and convergence to a pareto front. In: Paper presented at the Late breaking papers at the genetic programming 1998 conference
31. Wang J-J, Wang L (2020) A knowledge-based cooperative algorithm for energy-efficient scheduling of distributed flow-shop. *IEEE Trans Syst Man Cybern Syst* 50(5):1805–1819
32. Wen X, Wang K, Li H, Sun H, Wang H, Jin L (2021) A two-stage solution method based on NSGA-II for green multi-objective integrated process planning and scheduling in a battery packaging machinery workshop. *Swarm Evolut Comput* 61:100820
33. Wu X, Liu X, Zhao N (2019) An improved differential evolution algorithm for solving a distributed assembly flexible job shop scheduling problem. *Memet Comput* 11(4):335–355
34. Ying K-C, Lin S-W (2018) Minimizing makespan for the distributed hybrid flowshop scheduling problem with multiprocessor tasks. *Expert Syst Appl* 92:132–141
35. Zhang S, Yu Z, Zhang W, Yu D, Xu Y (2016) An extended genetic algorithm for distributed integration of fuzzy process planning and scheduling. *Math Probl Eng* 2016
36. Zhang S, Yu Z, Zhang W, Yu D, Zhang D (2015) Distributed integration of process planning and scheduling using an enhanced genetic algorithm. *Int J Innov Comput Inf Control* 11(5):1587–1602
37. Zhang X, Zhang H, Yao J (2020) Multi-objective optimization of integrated process planning and scheduling considering energy savings. *Energies* 13(23):6181
38. Zhang Z, Tang R, Peng T, Tao L, Jia S (2016) A method for minimizing the energy consumption of machining system: integration of process planning and scheduling. *J Clean Prod* 137:1647–1662
39. Zhao B, Gao J, Chen K, Guo K (2018) Two-generation Pareto ant colony algorithm for multi-objective job shop scheduling problem with alternative process plans and unrelated parallel machines. *J Intell Manuf* 29(1):93–108
40. Zhao F, Zhao L, Wang L, Song H (2020) An ensemble discrete differential evolution for the distributed blocking flowshop scheduling with minimizing makespan criterion. *Expert Syst Appl* 160:113678
41. Zitzler E, Deb K, Thiele L (2000) Comparison of multiobjective evolutionary algorithms: empirical results. *Evol Comput* 8(2):173–195

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.