Computers & Industrial Engineering 128 (2019) 458–476

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

An effective hybrid discrete grey wolf optimizer for the casting production scheduling problem with multi-objective and multi-constraint

Hongbin Qin, Pengfei Fan, Hongtao Tang*, Pan Huang, Bo Fang, Shunfa Pan

Hubei Key Laboratory of Digital Manufacturing, School of Mechanical and Electronic Engineering, Wuhan University of Technology, Wuhan 430070, China

ARTICLE INFO

Keywords:
Casting production scheduling
Processing interval constraint
Job transportation time
Initialization strategy
Grey wolf optimizer
Improved tabu search

ABSTRACT

Since there are some special constraints in a real foundry enterprise, including the limitation of starting time in some casting operations and the transportation time between two adjacent operations, processing interval constraint (PIC) and job transportation time (JTT) are introduced in this paper. With the consideration of PIC and JTT, a multi-objective casting production scheduling model is constructed to minimize makespan, the total production cost and the total delivery delay time. A hybrid discrete multi-objective grey wolf optimizer (HDMGWO) is developed to solve this model. An initialization strategy based on reducing job transportation time and processing time (RTP) are designed to improve the quality of initial population. A improved tabu search (ITS) algorithm is embedded into grey wolf optimizer (GWO) to overcome the premature convergence of the GWO. A modified search operator of GWO is designed to tackle discrete combinatorial optimization. A case example of the real foundry enterprise is illustrated to evaluate the effectiveness of proposed HDMGWO. Experimental results demonstrate that the proposed HDMGWO is superior in terms of the quality of solutions compared to five multi-objective algorithms. Real running in a casting ERP system verifies the applicability of the proposed scheduling model and the HDMGWO.

1. Introduction

With the deep integration of manufacturing and information technology, artificial intelligence is widely used in real-world production, of which the intelligent optimization algorithm is a typical (Hajiaghaei-Keshteli & Fathollahi-Fard, 2018; Kiani & Yildiz, 2016; Yıldız, 2017). As a application of intelligent optimization algorithm in manufacturing, scheduling plays an important role in production decisions, particularly in resource and energy-intensive industries (Jiang, Zheng, & Liu, 2018). The foundry is a typical resource and energy-intensive industry, which is eager to pursuit an effective solution of casting production scheduling problem (CPSP) to significantly improve the manufacturing cycle, rate of timely delivery and production cost.

The job-shop scheduling problem (JSP) has been proven to be an NP-hard (Blazewicz, Domschke, & Pesch, 1996), which can be expressed as allocating jobs to be processed in productive machines (Kato, Aranha, & Tsunaki, 2018). The flexible job-shop scheduling problem (FJSP) is an important extension of JSP, which allows one operation can be processed on one machine from a set of alternative machines and the processing time of each machine may not be the same. Hence, FJSP is a more complex NP-hard problem than JSP.

As a variant of the FJSP, the CPSP is less studied by scholars in comparison with other scheduling problems such as steelmaking scheduling issues (Li, Pan, Mao, & Suganthan, 2014; Pacciarelli & Pranzo, 2004; Tang, Zhao, & Liu, 2014), hot rolling scheduling issues (Chen, Yang, & Wu, 2008; Jia, Yi, Yang, Du, & Zhu, 2013) and welding scheduling problem (Lu, Gao, Li, & Xiao, 2017; Lu, Xiao, Li, & Gao, 2016). To sum up, the main reasons for the less research on the CPSP may be as follows:

(1) The production scheduling is the scheduling of machines and operations. However the degree of automation in foundry shop is not high, and there are non-machine processing operations in the manufacturing processes.
(2) Since most operations have their own processing machine in casting process, the number of machines involved in the scheduling is large. Moreover, the process route of casting is long and the process routes of different casings may not be the same. Those all make it harder to solve the CPSP.
(3) CPSP is more sophisticated in the practical production since it involves some realistic constraints, processing interval constraint and job transportation time. These constraints increase the complexity of formulating the mathematical modeling of CPSP.
(4) In the real-world casting production, the determination of a

* Corresponding author.
E-mail address: tanghongtaozc@163.com (H. Tang).

https://doi.org/10.1016/j.cie.2018.12.061
Received 12 August 2018; Received in revised form 13 December 2018; Accepted 25 December 2018
Available online 27 December 2018
0360-8352/

processing scheme often needs to take account of various performance indicators. Hence, CPSP is a discrete multi-objective combinatorial optimization (DMCO) problem.

The above reasons increase the complexity of this study. To cope with the problem, the main innovations of this paper are as follows:

(1) Developing a new mutl-objective scheduling model considering processing interval constraint, job transportation time simultaneously, among the first studies in this area.
(2) Proposing a new hybrid discrete multi-objective grey wolf optimizer to tackle CPSP firstly.
(3) Proposing a strategy that can reduce makespan and job transportation time to improve the quality of solution.
(4) Comparing the proposed algorithm with five multi-objective algorithms via three performance metrics for Pareto optimal solutions.
(5) Validation of the proposed algorithm through a real-world foundry enterprise case.

This paper is organized as follows: The brief literature review is provided in Section 2. Section 3 describes the problem and formulates a mathematical model for CPSP, while Section 4 sketches the original GWO. The proposed HDMGWO is presented in Section 5 and experimental studies are carried out in Section 6. Sections 7 gives a realistic application of the proposed HDMGWO. Finally, Section 8 presents conclusions and future work.

2. Literature review

The CPSP can be identified as a FJSP with complex constraints. From the relevant literature of FJSP, we summarize the three trends besides FJSP: multi-objective optimization, multi-constraint optimization and hybrid algorithm.

2.1. Multi-objective optimization

Makespan, which represents the longest completion time of the job, has always been the research focus of the single-objective FJSP (Khoukhi, Boukachour, & Alaoui, 2017; Li & Gao, 2016; Zhang, Gao, & Shi, 2011). Since the determination of a processing scheme in real-world production often needs to take account of various performance indicators, the multi-objective FJSP (MOFJSP) has been widely studied in recent years. Lu, Li, Gao, Liao, & Yi (2017) optimized the makespan and the total additional resource consumption through a new multi-objective discrete virus optimization algorithm. Nouiri et al. (2017) performed the scheduling under the machine breakdowns to optimize the makespan and the stability of the schedule. Zhang, Wang, and Liu (2017) proposed a dynamic game optimisation model for MOFJSP to minimize the makespan, the total workload of machines and the total energy consumption of production. Gong, Deng, Gong, Liu, and Ren (2018) formulated a scheduling model which considered machine flexibility and worker flexibility simultaneously, and then proposed a memetic algorithm to solve it whose objective is to minimise the makespan, workload of machines and the total workload of all machines.

Since it is hard to measure the fitness of a solution against multi-objective, a trade-off between multi-objective is needed. Kurdi (2017) summarized three approaches to address MOFJSP, including: weighting approach, lexicographical order-based approach and Pareto-based approach. Pareto-based approach makes it possible to achieve the final Pareto solution which has the better values in both objective functions (Fathollahi Fard & Hajiaghaei-Keshteli, 2018). Combining the multi-objective evolutionary algorithm (MOEA) with Pareto-based approach, the algorithm can search for multiple regions of the solution space and classify the solutions simultaneously in the iteration, to get multiple solutions of the problem in single run (Pholdee, Bureerat, & Yıldız, 2017). Wang, Zhou, Xu, and Liu (2012) presented an improved pareto-based artificial bee colony algorithm to address MOFJSP efficiently. Yuan and Xu (2015) developed a memetic algorithm to solve MOFJSP flexibility, which added a novel local search to NSGA-II. Gao et al. (2014) solved MOFJSP with a pareto-based grouping discrete harmony search algorithm.

2.2. Multi-constraint optimization

A survey of FJSP is provided by Chaudhry and Khan (2016), which showed that most of work on the FJSP focused on addressing simple FJSP rather than considering different scenarios. In recent years, there has been a growing interest in the FJSP to consider different application scenarios. Chung, Sun, and Liao (2017) developed two immunoglobulin-based artificial immune system algorithms for a two-stage hybrid flowshop problem with waiting time constraint between two stages. Lu, Li, et al. (2017) investigated the FJSP with controllable processing times. They proposed a multi-objective discrete virus optimization algorithm (MODVOA) to minimize the makespan and the total additional resource consumption and compared MODVOA with NSGA-II and SPEA2. Moreover, a realistic production scheduling case was presented to verify the ability of the proposed MODVOA to solve real-world production scheduling problems. Nouiri et al. (2017) considered the FJSP under unexpected disruptions such as machine breakdowns. A two-stage particle swarm optimization was presented to minimize the makespan and stability. Meng, Pan, and Sang (2018) proposed a hybrid artificial bee colony (hyABC) algorithm to tackle FJSP with overlapping in operations. In proposed hyABC, a modified migrating birds optimisation algorithm was integrated into the artificial bee colony algorithm to perform search process. Khoukhi et al. (2017) proposed a novel hybrid Ant Colony Optimization approach with dynamic history for the FJSP with machine unavailability constraints. To describe the proposed mathematical model clearly, a bi-level disjunctive/conjunctive graph was introduced.

More and more research papers consider the different scenarios of FJSP. But most of the works are focused on testing the proposed algorithm in the theoretical way, such as benchmark or generated problems (Chaudhry & Khan, 2016). Hence, it is necessary to validate the proposed algorithm through a real-world case.

2.3. Hybrid algorithms

Hybrid algorithms which take advantage of the strengths of each of the algorithms have received significant interest for fast convergence and robustness in searching better solutions (Yildiz, 2012, 2013a). Fathollahi-Fard, Hajiaghaei-Keshteli, and Mirjalili (2018a) proposed a tri-level decision-making model for tire closed-loop supply chain. Four new hybrid algorithms based on four metaheuristics were developed to tackle the problem and the hybrid Keshtel Algorithm and Simulated Annealing performed best. Yıldız and Lekesiz (2017) presented a hybrid charged system search and Nelder–Mead algorithm (HCSSNM) for the optimisation of a vehicle suspension arm and compared the proposed HCSSNM with genetic algorithm, Gravitational search algorithm and charged system search. Gen, Zhang, Lin, and Yun (2017) summarized three hybrid evolutionary algorithms and demonstrated them to five scheduling problems in manufacturing systems.

Metaheuristics have been proven their ability to solve combinatorial optimization problem (Fathollahi-Fard & Hajiaghaei-Keshteli, 2018; Nouiri, Bekrar, Jemai, Niar, & Ammari, 2018; Yildiz & Saitou, 2011; Yildiz, 2013b; Zhang, Gao, & Li, 2013). As a Metaheuristics, the grey wolf optimizer (GWO) proposed by Mirjalili, Mirjalili, and Lewis (2014) is inspired by hunting for the prey of grey wolves. Compared with other algorithms such as particle swarm optimization, genetic algorithm and gravitation search algorithm, GWO performs better both in solution quality and stability (Faris, Aljarah, Al-Betar, & Mirjalili, 2018). The GWO has been successfully applied to many engineering fields, such as image thresholding (Li et al., 2016), Optimal Reactive Power Dispatch

Problem (Nuaekaew, Artrit, Pholdee, & Bureerat, 2017) and welding scheduling (Lu et al., 2016). Since GWO is suitable to tackle the continuous function problem, there are few applied researches on the production scheduling which is a more complex discrete combinatorial optimization problem (DCOP). And the existing literatures about production scheduling focus only on the flow shop scheduling problem (Komaki & Kayvanfar, 2015; Lu et al., 2016; Lu, Li, et al., 2017), which is not suitable to address CPSP. Therefore, in this paper, a new hybrid discrete multi-objective GWO with Tabu Search (TS) is proposed to tackle CPSP. TS has a good exploration ability, the hybrid of GWO and TS can make the proposed algorithm have better exploration and exploitation ability simultaneously. The above reasons motivate us to develop a hybrid discrete multi-objective GWO with TS for this CPSP.

The purpose of this paper is to propose an effective multi-objective predictive-reactive scheduling method for the CPSP to narrow the gap between theoretical research and real-world application. In this paper, two constraints inspired in a real foundry enterprise are introduced. Then, a multi-objective casting production scheduling model considering above two constraints is formulated to minimize makespan, the total production cost and the total delivery delay time. In addition, a new hybrid discreet multi-objective grey wolf optimizer is proposed to solve this model. In proposed algorithm, a new initialization strategy is proposed to improve the solution. A improved tabu search is embedded into GWO to perform local search. Besides, a modified search operator for discrete combinatorial optimization is designed to update the positions of the grey wolf individual. Finally, a real case of the CPSP is illustrated to demonstrate the superiority of the proposed algorithm over five multi-objective algorithms. And a realistic application is presented to verify the applicability of the proposed scheduling model and the algorithm.

3. Problem description and mathematical modeling

In this section, we give a detail description of CPSP. Then, a multi-objective mathematical model for the CPSP which derives from a realistic foundry shop is formulated.

3.1. Problem description

Casting is a process where a liquid metal is poured into a casting cavity adapted to the shape of the part, then obtain a part or a blank after cooling and solidifying. The casting process usually consists of three parts: mold making, melting and pouring, post-processing. Mold making denotes making a suitable container that can change liquid metal into blank after cooling. Melting and pouring means that the raw materials are melted into liquid metal and poured into the mold. Post-processing is a series of operations on the blank to get the product that meets the requirements. The flow chart of casting process in a foundry shop is given in Fig. 1. It can be seen that each operation has its own set of alternative machines. Some special operations like degating, weld repairing, grinding and flaw detection are processed by workers rather than machines. Unlike classic FJSP, the processing between two adjacent operations in CPSP is not continuous. The next operation of the job cannot be processed until the previous operation of the job is finished and transported to the idle machine processing the next operation. The significant difference between CPSP and classic FJSP is that the operations in CPSP like melting and heat treatment have strict processing interval constraint which must be processed within the specified time interval. The long processing time and large power consumption result in the high cost of electricity in melting and heat treatment. Therefore, the melting furnace and heat-treatment furnace will only operate at night to reduce costs (electricity rate is cheaper at night in China).

Based on the classic FJSP, this paper proposes the notions of virtual operation and virtual working hours, processing interval constraint and job transportation time, which are inspired from the real-world foundry shop. The related notions are defined as follows:

Definition 1: virtual operation and virtual working hours (VOVH). In this paper, the workers are considered as machines to perform scheduling. The virtual operation denotes that the operations processed by workers are treated as by machines and the virtual working hours represents the processing hours of the operations processed by workers. The CPSP can be transformed into a FJSP after introducing the notions of virtual operation and virtual working hours.

Definition 2: processing interval constraint (PIC). If the operation has strict processing interval constraint, this operation is called processing constraint operation (PCO). The constraint interval of the PCO is represented by PI_{ij}. 24-h system is adopted in this paper. For example, $PI_{ij} = [9,17]$ denotes that the operation O_{ij} must be processed between 9:00 on the day and 17:00 on the same day, and $PI_{ij} = [22,6^{+1}]$ denotes that O_{ij} must be processed between 22:00 on the day and 6:00 on the next day.

Definition 3: job transportation time (JTT). In this paper, the JTT needs to be considered when the job is transported between two adjacent operations. If the two adjacent operations of the job are processed in the same workshop, JTT can be ignored; otherwise, JTT should be considered which is determined by machines selected by the two adjacent operations of the job. Moreover, there are enough transporters to transport the job from one machine to another. The loading and unloading times are included in the JTT.

The general description of the CPSP considering the VOVH, PIC and the JTT is as follows: jobs $\{J_1, J_2, \ldots, J_n\}$ are processed on machines $\{M_1, M_2, \ldots, M_m\}$. Each job J_i contains multiple operations, and the number of operations in different job may be differernt. Each job is processed according to the specified process route and n_i is the total number of operations in J_i. Each operation can be processed by one machine from a set of alternative machines $M_{ij} \subseteq \{1,2,\ldots,M\}$, and the processing time of different machines are not necessarily the same. The JTT must be considered if two adjacent operations are processed in different workshops. For special operation with PIC, the processing time of the operation must be in PI. The aim of CPSP is to choose a suitable processing machine for each operation, and select a reasonable processing sequence for each machine to optimize three indicators: makespan, the total production cost and the total delivery delay time.

This CPSP is based on the following assumptions: (1) At time 0, all machines are available and all jobs can be processed in the first operation; (2) Each operation can only be processed on one machine; (3) Each machine can only process one operation of the job at a certain time, and each machine can hold an unlimited number of jobs to be processed; (4) There is no processing sequence constraint between different jobs, but there is processing sequence constraint between different operations of the job; (5) The sequence of all operations of the job cannot be changed, the job cannot be transported to the next operation and start the next operation until the current operation is completed; (6) Ignore the JTT before the first operation and after the last operation; (7) As soon as the operation is completed, the job is immediately transported to the next operation, regardless of batch transportation; (8) All machines are started before processing, regardless of machine setup time; (9) All machines keep running until all jobs are completed.

3.2. Mathematical modeling for CPSP

To formulate a three-objective mathematical model for CPSP, we introduce some parameters, decision variables and constraints:

3.2.1. Parameters

J_i	jobs need to be processed	ST_{ijk}	starting time of O_{ij} on M_k
n	the number of jobs	C_{ijk}	completion time of O_{ij} on M_k
n_i	the number of operations in J_i	$O_{i'j'}$	one operation on M_k
m	the number of machines	C_i	completion time of J_i
M	machine set	N	a very large positive number
M_k	kth machine	RC_i	raw material cost of J_i
O	job set	F_k^v	dynamic processing rate of M_k
O_{ij}	jth operation in J_i	F_k^s	static waiting rate of M_k
M_{ij}	the set of alternative machines of O_{ij}	T_k^v	dynamic processing time of M_k
P_{ijk}	processing time of O_{ij} on M_k	T_k^s	static waiting time of M_k
t_{kl}	JTT from kth machine to lth machine	VC_k	dynamic processing cost of M_k
D_i	delivery time of J_i	SC_k	static waiting cost of M_k
PI_{ij}	PIC of O_{ij}	TC	transportation rate
π	a feasible solution	Del_{ijk}	the delay time of O_{ij} on M_k

3.2.2. Decision variables

$$x_{iji'j'} = \begin{cases} 1, & \text{if } O_{ij} \text{ is processed before } O_{i'j'} \\ 0, & \text{if } O_{i'j'} \text{ is processed before } O_{ij} \end{cases}$$

$$Z_{iji'j'k} = \begin{cases} 1, & \text{if } O_{ij} \text{ is processed before } O_{i'j'} \text{ on } M_k \\ 0, & \text{if } O_{i'j'} \text{ is processed before } O_{ij} \text{ on } M_k \end{cases}$$

$$V_{ijk} = \begin{cases} 1, & \text{if } O_{ij} \text{ is processed on } M_k \\ 0, & \text{otherwise} \end{cases}$$

$$S_{ij} = \begin{cases} 1, & \text{if } O_{ij} \text{ is PCO} \\ 0, & \text{otherwise} \end{cases}$$

ST_{ijk},: the starting time of O_{ij} on M_k. If O_{ij} is not the first operation on M_k and is not the PCO, there are two possible results of ST_{ijk} which are shown in Fig. 2 to illustrate. In Result 1, M_k is processing $O_{i'j'}$ when O_{ij} is transported to M_k. Therefore, O_{ij} cannot be processed on M_k until $O_{i'j'}$ is finished. Result 2 denotes that M_k is idle when O_{ij} is transported to M_k. When O_{ij} is not the first operation on M_k and is the PCO, there are seven possible results of ST_{ijk} which are shown in Fig. 3. In this case, ST_{ijk} is relevant to the maximum value of $C_{i'j'k}$ and $C_{i(j-1)k}$ plus t_{lk}, min(PI_{ij}) and max(PI_{ij}). In conclusion, the ST_{ijk}, should meet the following conditions:

$$ST_{ijk} = \begin{cases} 0, & \text{if } O_{ij} \text{ is the first operation on } M_k \\ NST_{ijk}, & \text{if } O_{ij} \text{ is not the PCO and processed after } O_{i'j'} \text{ on } M_k \\ CST_{ijk}, & \text{if } O_{ij} \text{ is the PCO and processed after } O_{i'j'} \text{ on } M_k \end{cases}$$

where NST_{ijk} and CST_{ijk} are formulated as follows:

$$NST_{ijk} = \max(C_{i'j'k}, C_{i(j-1)l} + t_{lk})$$

$$CST_{ijk} = \begin{cases} \min(PI_{ij}), & \text{if } NST_{ijk} \leqslant \min(PI_{ij}) \\ ES_{i'j'i(j-1)}, & \text{if } \min(PI_{ij}) < NST_{ijk} < \max(PI_{ij}) \\ \min(PI_{ij}) + 24, & \text{if } NST_{ijk} \geqslant \max(PI_{ij}) \end{cases}$$

3.2.3. Objective functions

With above parameters, the CPSP can be described as a mixed integer programming (MIP) model to minimize the makespan, the total production cost and total delivery delay time. There objective functions of CPSP are formulated as follows.

(1) **Makespan**

The makespan refers to the longest job completion time, which is denoted by f_1 and can be formulated as follows:

$$\min f_1 = \max(C_i) \quad (1)$$

where C_i is the completion time of J_i, which consists of three parts: the total processing time of J_i, the total transportation time of J_i and the total delay time of J_i. The total processing time of J_i and the total transportation time of J_i are only affected by machine selection, but the total delay time of J_i is not only affected by the machine selection but also by the processing sequence on the machine and PIC. C_i can be formulated as follows:

$$C_i = \sum_{j=1}^{n_i} P_{ijk} + \Sigma^m t_{lk} + \sum_{j=1}^{n_i} Del_{ijk} \quad (2)$$

where Del_{ijk} is the delay time of O_{ij} on M_k. There are three possible situations. In situation 1, O_{ij} is the first operation on M_k, which represents that there is no delay time of O_{ij} on M_k. Situation 2 is that O_{ij} is not the PCO and is processed after $O_{i'j'}$ on M_k. Two probable results of Del_{ijk} in this situation are shown in Fig. 2. Situation 3 represents that O_{ij} is the PCO and is processed after $O_{i'j'}$ on M_k, the seven probable results of which is presented in Fig. 3. Therefore, Del_{ijk} can be formulated as follows:

$$Del_{ijk} = \begin{cases} 0, & \text{if } O_{ij} \text{ is the first operation on } M_k \\ NDel_{ijk}, & \text{if } O_{ij} \text{ is not the PCO and processed after } O_{i'j'} \text{ on } M_k \\ CDel_{ijk}, & \text{if } O_{ij} \text{ is the PCO and processed after } O_{i'j'} \text{ on } M_k \end{cases} \quad (3)$$

where $NDel_{ijk}$ and $CDel_{ijk}$ are formulated as follows:

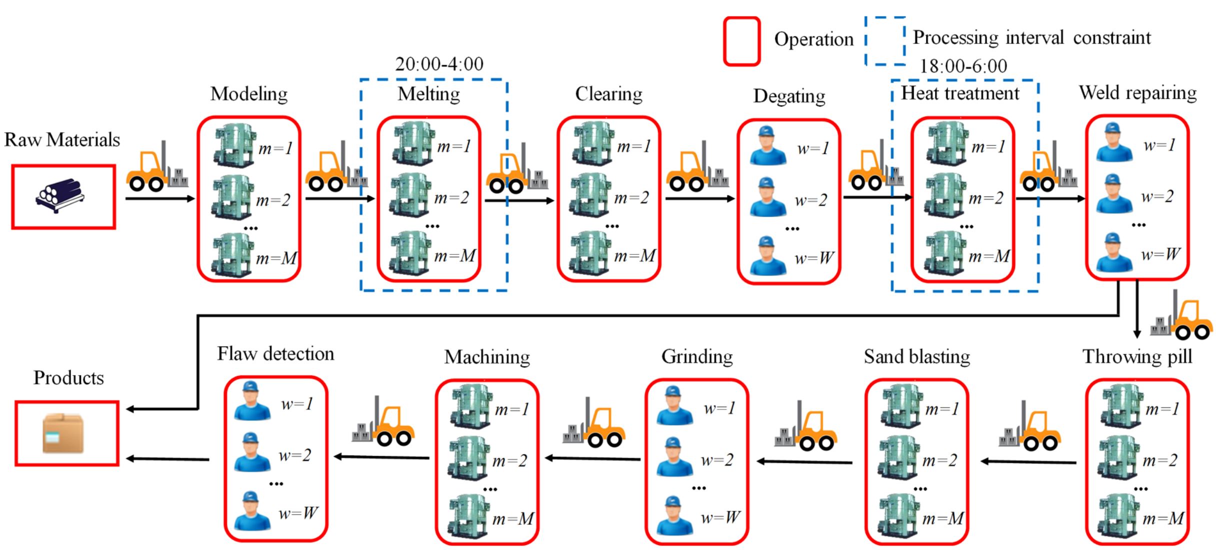

Fig. 1. The flow chart of casting process in a foundry shop.

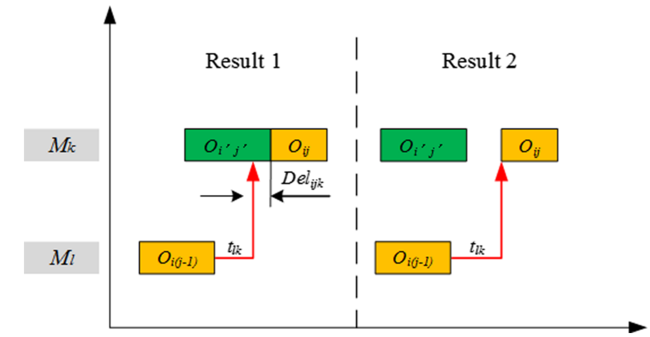

Fig. 2. Two possible results of ST_{ijk} when O_{ij} is not the first operation on M_k and is not the PCO.

$$NDel_{ijk} = \begin{cases} 0, & \text{if } C_{i'j'k} \leqslant C_{i(j-1)l} + t_{lk} \\ C_{i'j'k} - (C_{i(j-1)l} + t_{lk}), & \text{if } C_{i'j'k} > C_{i(j-1)l} + t_{lk} \end{cases} \tag{4}$$

$$CDel_{ijk} = \begin{cases} \min(PI_{ij}) - (C_{i(j-1)l} + t_{lk}), \quad \text{if } NST_{ijk} \leqslant \min(PI_{ij}) \\ C_{i'j'k} - (C_{i(j-1)l} + t_{lk}), \\ \quad \text{if } \min(PI_{ij}) < C_{i'j'k} < \max(PI_{ij}) \text{ and } C_{i'j'k} \geqslant C_{i(j-1)l} + t_{lk} \\ 0, \quad \text{if } \min(PI_{ij}) < C_{i(j-1)l} + t_{lk} < \max(PI_{ij}) \text{ and } C_{i(j-1)l} + t_{lk} \leqslant C_{i'j'k} \\ \min(PI_{ij}) + 24 - (C_{i(j-1)l} + t_{lk}), \quad \text{if } NST_{ijk} \geqslant \min(PI_{ij}) \end{cases} \tag{5}$$

(2) **Production cost**

In this paper, the processing rate denotes the cost per unit time when machine is working. And it can be divided into two categories, static waiting rate and dynamic processing rate. Static waiting rate represents the cost per unit time when machine is in the idle run. And dynamic processing rate denotes the cost per unit time when machine is in processing state. The manufacturing cost of the job can be calculated by multiplying the processing rate by the corresponding consumption matrix. The production cost denoted by f_2 is the sum of manufacturing cost, the raw material cost and the transportation cost, which can be formulated as follows:

$$\min\ f_2 = \sum_{k=1}^{m} (VC_k + SC_k) + \sum_{i=1}^{n} RC_i + TC\cdot \sum_{M} t_{kl} \tag{6}$$

where VC_k and SC_k are formulated as follows:

$$VC_k = \sum_{k=1}^{m} (T_k^v \cdot F_k^v) \tag{7}$$

$$SC_k = \sum_{k=1}^{m} (T_k^s \cdot F_k^s) \tag{8}$$

(3) **Delivery delay time**

The total delivery delay time denoted by f_3 can be formulated as follows:

$$\min\ f_3 = \sum_{i=1}^{n} (C_i - D_i) \tag{9}$$

3.2.4. Constraints

$$S_{i1k} = 0, \quad i \in n;\ k \in M_{i1} \tag{10}$$

$$\sum_k V_{ijk} = 1, \quad i \in n;\ j \in n_i;\ k \in M_{ij} \tag{11}$$

$$C_{ijk} \leqslant ST_{i(j+1)l}, \quad i \in n;\ j, (j+1) \in n_i;\ k \in M_{ij};\ l \in M_{i(j+1)} \tag{12}$$

$$C_i \geqslant \sum_{j=1}^{n_i} C_{ij}, \quad i \in n;\ j \in n_i \tag{13}$$

$$C_{ijk} \geqslant ST_{ijk} + P_{ijk} - N(1 - V_{ijk}), \quad i \in n;\ j \in n_i;\ k \in M_{ij} \tag{14}$$

$$ST_{ijk} \geqslant C_{i'j'k} - N\cdot Z_{iji'j'k} \tag{15}$$

$$ST_{i'jk'} \geqslant C_{ijk} - N\left(1 - Z_{iji'j'k}\right) \tag{16}$$

$$N\left(1 - x_{ijj'}\right) + ST_{ij'l} \geqslant ST_{ijk} + P_{ijk} + t_{kl} \tag{17}$$

$$N\cdot x_{ijj'} + ST_{ijk} \geqslant ST_{ij'l} + P_{ij'}l + t_{lk} \tag{18}$$

$$N(1 - S_{ij}) + ST_{ijk} \geqslant \min(PI_{ij}) \tag{19}$$

$$N\cdot S_{ij} + ST_{ijk} \geqslant \max(PI_{ij}) \tag{20}$$

Eq. (10) represents assumption 1; Eq. (11) represents assumption 2; Eq. (12) ensures that the processing sequence between different operations of the job is specified, that is, assumption 4; Eq. (13) determines the completion time of the job; Eq. (14) denotes that the difference between completion time and starting time is no less than the processing time on M_k, that is, assumption 2; Constraints (15)–(16) guarantee that two operations can't be processed on the same machine at the same time, that is, assumption 3; Constraints (17)–(18) represents assumption 5; Constraints (19)–(20) restrict that the processing time of the PCO should be in *PI*.

4. The original GWO

As a metaheuristic based on swarm intelligence, GWO was proposed by mimicking the social hierarchy and predatory behavior of gray wolves. In GWO, the gray wolf individuals are divided into four categories, the best wolves are treated as alpha wolf (α), beta wolf (β) and delta wolf (δ) while the rest are omega wolf (ω). The predatory behavior in GWO can be divided into three major stages: encircling prey, hunting and attacking prey.

To model encircling behavior of grey wolves, the equation is defined as follows:

$$D = |C\cdot X_P(t) - X(t)| \tag{21}$$

$$X(t+1) = X_P(t) - A\cdot D \tag{22}$$

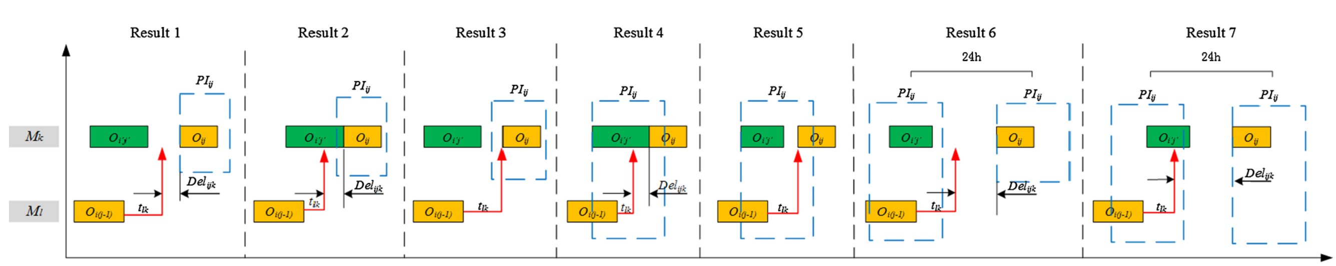

Fig. 3. Seven possible results of ST_{ijk} when O_{ij} is not the first operation on M_k and is the PCO.

$$A = 2a \cdot r_1 - a \tag{23}$$

$$C = 2 \cdot r_2 \tag{24}$$

$$a = 2 - t \cdot (2/t_{\max}) \tag{25}$$

where t denotes the current iteration, t_{max} represents the maximum iteration. Vector A and C are adaptive vectors. X_P represents the position vector of the prey and X represents the position vector of the wolf. Vector a decreases linearly from 2 to 0. r_1 and r_2 are random number vectors in [0,1].

The grey wolves will hunt after encircling prey. In this process, the other wolves (ω) update their positions according to the guidance of α, β and δ. The equation of hunting is formulated as follows:

$$D_\alpha = |C_1 \cdot X_\alpha(t) - X(t)|,\ \ D_\beta = |C_2 \cdot X_\beta(t) - X(t)|,\ \ D_\delta = |C_3 \cdot X_\delta(t) - X(t)| \tag{26}$$

$$X_1 = X_\alpha - A_1 \cdot D_\alpha,\ \ X_2 = X_\beta - A_2 \cdot D_\beta,\ \ X_3 = X_\delta - A_3 \cdot D_\delta \tag{27}$$

$$X(t+1) = (X_1 + X_2 + X_3)/3 \tag{28}$$

The last stage is attacking the prey, that is, the GWO obtains the optimal solution. The grey wolves diverge from each other to search for the prey when $A > 1$ and converge to attack the prey when $A < 1$. In order to see more detailed description about the GWO, interested readers are referred to: Mirjalili et al. (2014).

5. The proposed HDMGWO for CPSP

The original GWO is developed mainly to tackle the continuous optimization problem. However, the CPSP is a DMCO problem. In order to improve the applicability of GWO to solve DMCO and address its drawbacks in local search, a new multi-objective discrete metaheuristic based on the GWO is proposed to solve CPSP. In proposed HDMGWO, a discrete multi-objective grey wolf optimizer is designed in global search stage and a improved tabu search is designed in local search stage. The basic flow chart of proposed HDMGWO is presented in Fig. 4. The details of steps can be described as follows:

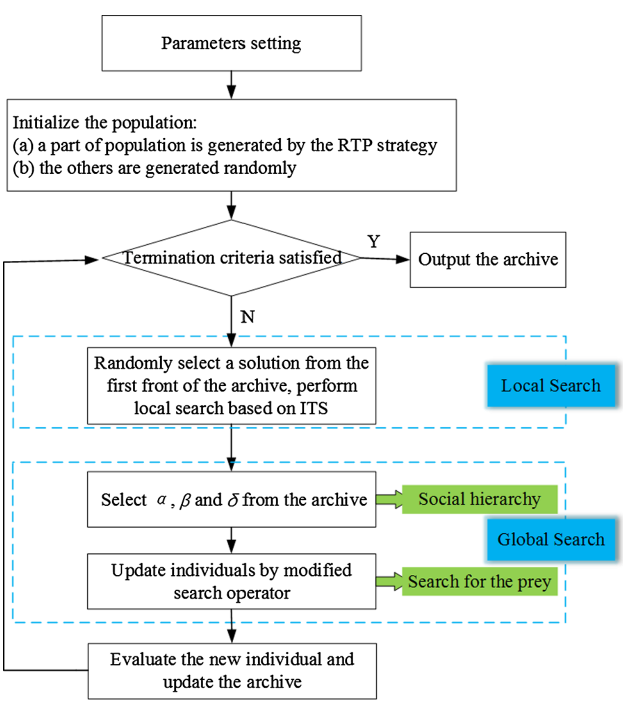

Fig. 4. The flow chart of the proposed HDMGWO.

Step 1: Set parameters.
Step 2: Initialize the population (see Section 5.2 for details).
Step 3: Is the termination criteria satisfied? If yes, output the archive; else, perform **Step 4**.
Step 4: Evaluate the initial solutions by fast non-dominated sorting proposed by Deb, Pratap, Agarwal, and Meyarivan (2002) and update archive. Randomly select a solution from the first front of the archive to perform local search based on ITS (see Section 5.3 for details) and update the archive.
Step 5: Select three best solutions from the archive and set as α, β and δ respectively (see Section 5.4.1 for details).
Step 6: Update the position of the individual by the modified search operator (see Section 5.4.2 for details).
Step 7: Evaluate the new individual and update the archive, then perform **Step 3**.

The following subsections give a detailed explanation about the improvement procedures of HDMGWO, including: encoding and decoding, initialization by RTP strategy, local search by improved tabu search and global search by social hierarchy and modified search operator.

5.1. Encoding and decoding

An appropriate encoding and decoding scheme can affect the performance of the algorithm and make it easier for the algorithm to generate feasible solution. The main idea of the original GWO is to obtain the optimal solution by updating the position of the grey wolf individual which is a continuous variable. However, the CPSP is a typical discrete problem. The original GWO algorithm cannot directly update the operation sequence of the job. Therefore, it is a essential issue to solve CPSP by mapping the continuous position of the grey wolf individual in the original GWO to the discrete scheduling of the job in the CPSP.

The CPSP has to deal with the operation sequence and the machine assignment simultaneously. Thus, the encoding scheme of CPSP contains two parts. The first part is operation sequence denoted by X_O and the second part is machine assignment denoted by X_m. The length of two parts are both D which is the sum of the operations of all jobs and they constitute a feasible solution (the position of grey wolf individual). Tables 1 and 2 are the instance of CPSP with PIC and JTT. The symbol *inf* represents that the job can't be transported between two machines.

One of the feasible solutions in this problem is $\pi = [1,1,3,3,1,2,2,3,2,1,3,1,2,2,2,1,1,1]$, where $X_o = [1,1,3,3,1,2,2,3,2]$ and $X_m = [1,3,1,2,2,2,1,1,1]$. The encoding scheme for this solution is shown in Fig. 5. The element value in X_o represents the job number, and the times this element appears represent the operation number of the job.

Table 1
The processing time and PIC of the operation.

Job	Operation	*PI*	Processing time (h)						
			M_1	M_2	M_3	M_4	M_5	M_6	M_7
Job 1	O_{11}	–	8	6	–	–	–	–	–
	O_{12}	$[20,6^{+1}]$	–	–	4	6	5	–	–
	O_{13}	–	–	–	–	–	–	5	3
Job 2	O_{21}	–	7	6	–	–	–	–	–
	O_{22}	$[20,6^{+1}]$	–	–	4	4	5	–	–
	O_{23}	–						5	6
Job 3	O_{31}	–	6	9	–	–	–	–	–
	O_{32}	$[20,6^{+1}]$	–	–	6	3	5	–	–
	O_{33}	–	–	–	–	–	–	3	4

Table 2
The JTT between two machines.

Machines	JTT (h)						
	M_1	M_2	M_3	M_4	M_5	M_6	M_7
M_1	*inf*	*inf*	1	1	2	*inf*	*inf*
M_2	*inf*	*inf*	2	1	3	*inf*	*inf*
M_3	*inf*	*inf*	*inf*	*inf*	*inf*	2	1
M_4	*inf*	*inf*	*inf*	*inf*	*inf*	1	3
M_5	*inf*	*inf*	*inf*	*inf*	*inf*	3	2
M_6	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*
M_7	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*

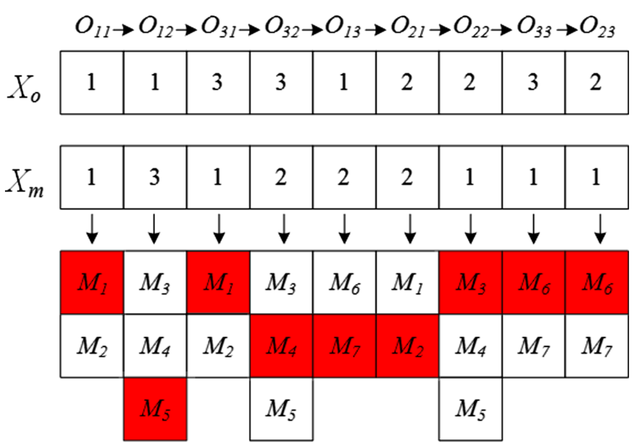

Fig. 5. The encoding scheme.

Such as the second element in X_o is 1, which appears twice, denoting the second operation of job 1 (O_{12}). The element of X_m indicates the index of the machine selected in the set of alternative machines of the corresponding operation. Such as the second element in X_0 is 3, which denotes selecting the third machine in alternative machines set $M_{12} = \{M_3, M_4, M_5\}$ to process O_{12}.

Decoding scheme can be described as the reverse of encoding. Since the processing interval constraint exists in this scheduling, the processing time of PCO must meet Constraints (19)–(20). The Gantt chart of the feasible solution π obtained by the decoding scheme is given in Fig. 6.

5.2. Initialization by RTP strategy

In order to improve the quality of initial population, RTP strategy is designed in this paper, which can reduce transportation time and processing time of the job simultaneously. In addition, to improve the diversity of initial population, a random initialization strategy is also adopted. RTP strategy uses a multi-stage decision approach by establishing a disjunctive graph model of transportation time. The detailed explanation about the RTP strategy is as follows:

First, we establish a disjunctive graph model of transportation time. A transportation time disjunctive graph of CPSP with 3 jobs and 10 operations is given in Fig. 7. The arc of the disjunctive graph has a weight value which represents the transportation time between two operations. Taking job 2 with 4 operations for example, O_{21} and O_{24} have two alternative machines. According to assumption 6, (0,0) represents the transportation time from the materials database to the machine processing O_{21}. Simultaneously, (0,0) represents the transportation time from the machine processing O_{24} to the warehouse. O_{22} has two alternative machines that form a 2×2 transportation time matrix with the two alternative machines of O_{21}. The transportation time matrix of the subsequent operations are similarly.

We further analyze the transportation process of job 2 and establish a directed acyclic graph (see Fig. 8). In Fig. 8, the vertices represents the alternative machines of the operation, such as M_1 and M_2 are the two alternative machines of O_{21}. The arc of the directed acyclic graph has a weight value which represents the transportation time between machines. Obviously, there are 24 paths in Fig. 8. The shortest transportation time in the 24 paths is the optimal machine selection scheme for the job 2.

Then, the information in Fig. 8 can be represented as a transportation time array (see Fig. 9). The number of rows in the array indicates the outgoing operation, and the number of columns indicates the incoming operation. Due to fact that each operation has a set of alternative machines, there are transportation time matrices in the array. The symbol *inf* represents that the job can't be transported between two operations.

Finally, the processing time of each operation is assigned to the node in Fig. 8 and the directed acyclic graph is divided into 4 stages

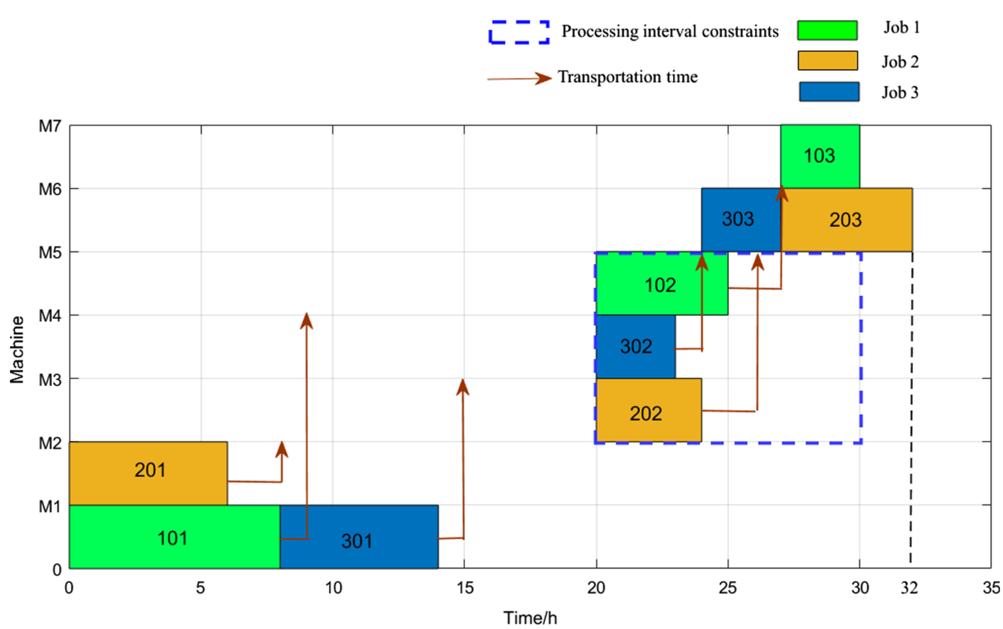

Fig. 6. The Gantt chart of the solution after decoding.

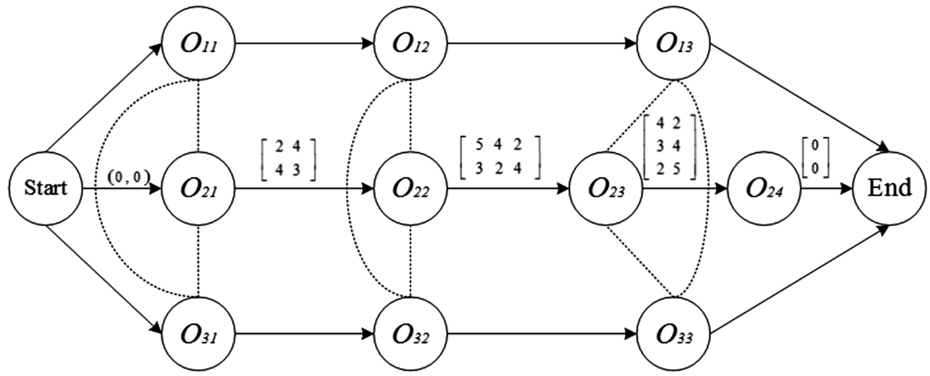

Fig. 7. The disjunctive graph of transportation time.

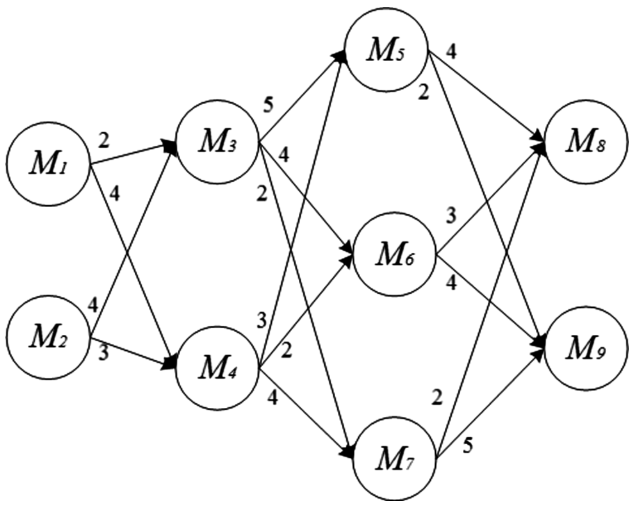

Fig. 8. The directed acyclic graph of transportation process.

$$\left\{ \begin{array}{cccc} inf & \begin{array}{c} 2\ 4 \\ 4\ 3 \end{array} & inf & inf \\ inf & inf & \begin{array}{c} 5\ 4\ 2 \\ 3\ 2\ 4 \end{array} & inf \\ inf & inf & inf & \begin{array}{c} 4\ 2 \\ 3\ 4 \\ 2\ 5 \end{array} \\ inf & inf & inf & inf \end{array} \right\}$$

Fig. 9. The array of transportation time.

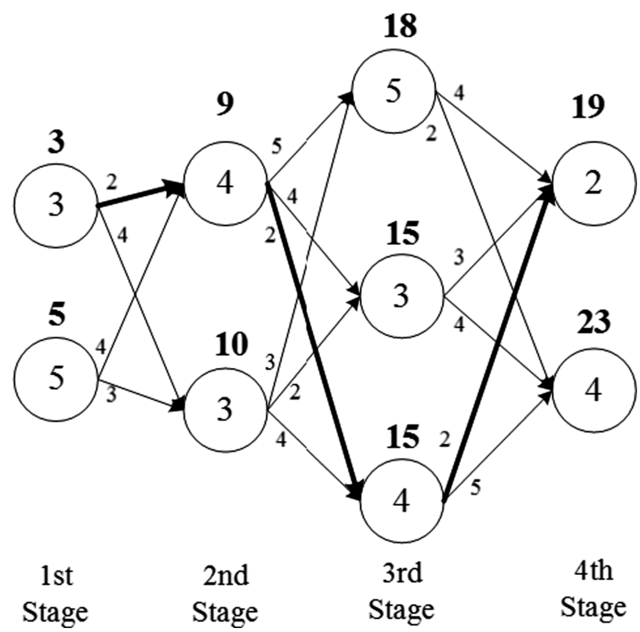

Fig. 10. The directed acyclic graph with out-degree.

according to the operations of job 2 (see Fig. 10). The in-degree and out-degree of each node is calculated separately, where in-degree is the sum of the out-degree and arc weights of the node in the previous stage, and the out-degree is the sum of the processing time of the node and the in-degree. The RTP strategy uses a multi-stage decision approach which selects the machine with minimum out-degree at each stage. In Fig. 10, the out-degree of each node is marked in bold. The 3rd stage exists two minimum out-degree: Path 1: M_1–M_3–M_7–M_8 is the optimal path (marked in bold in Fig. 10) which takes 19 h; Path 2: M_1–M_3–M_6–M_8 takes 20 h. Although the selected path by RTP strategy may not be the true optimal path of the job, it is also the better path. Consequently, the RTP strategy can play the role of reducing the job transportation time and processing time simultaneously, and even can improve the diversity of the initial solution.

5.3. Local search by improved tabu search

Due to the fact that the searching process of GWO is guided by α, β and δ, the GWO tends to fall into the local optimal solution. In this paper, the TS is embedded into GWO to overcome the premature convergence of the GWO. The TS proposed by Glover (1990) has been widely used to tackle combinatorial optimization problems (Gendreau, Iori, Laporte, & Martello, 2006; Li, Pan, & Liang, 2010; Liu, Yi, Wu, Ye, & Chen, 2008). The original TS has a large search neighborhood and its searching process takes a long time. Thus, a improved tabu search is designed in this paper to perform local search more efficiently. The main improvement of improved tabu search consists of two parts: the dynamic adjustment mechanism and hybrid search strategy, which are introduced below.

5.3.1. The dynamic adjustment mechanism

In original TS, the parameters like tabu list length and neighborhood solutions size are fixed and cannot be tuned adaptively with the searching process, which cannot ensure that the TS to be performed

```
procedure: dynamic adjustment mechanism
input:current iteration It, maximum iteration MaxIt,
        operation number k, job number Jnumber
output: neighborhood solutions size temp_nb,
          tabu list length tabu_list_length
begin
        delta_t= MaxIt/5
        min_nb=5Jnumber
        max_nb=10Jnumber
        temp_nb=ceil(min_nb+(max_nb-min_nb)*(It/MaxIt))
        if   It < 2 delta_t
             tabu_list_length=round(k/2)
        if   It < 4 delta_t
             tabu_list_length=round(k*((3It)/(4delta_t)-1))
        if   It < 5 delta_t
             tabu_list_length=2k
end
```

Fig. 11. Pseudo-code of dynamic adjustment mechanism.

efficiently in the whole searching process. Thus, a dynamic adjustment mechanism of tabu list length and neighborhood solutions size are developed in this paper. In the initial search, the tabu list length and neighborhood solutions size should be small, which can expand the search space and improve the dispersion of the solution. On the contrary, when the search process is close to the optimal solution, the larger tabu list length and neighborhood solutions size can narrow the search space and improve the convergence of the solution. The dynamic adjustment mechanism of tabu list length and neighborhood solutions size for the current iteration are given in Fig. 11.

5.3.2. Hybrid search strategy

The selection of neighborhood structure has a great influence on the solution quality and execution efficiency of the TS. The original TS searches for the neighborhood solutions around the current solution randomly, which will result in blindness of the search. The use of neighborhood structure based on critical path can reduce the space for local search and improve search efficiency. Related researches (Nowicki & Smutnicki, 1996; Sha & Hsu, 2006; van Laarhoven, Aarts, & Lenstra, 1992) show that swapping internal operations within a critical block cannot reduce the maximum makespan. In this paper, a hybrid search strategy consists of better search and random search are developed, which are introduced below.

(1) **better search**

The better search consists of four effective neighborhood structures based on moving the front or the rear of the critical block. Suppose that a critical block $CB = \{CB_1, CB_2, \ldots, CB_{n-1}, CB_n\}$, where $CB_1, \ldots, CB_n$ are the critical operations. First, we define three functions: (1) *Swap* (x,y) denotes swapping x with y; (2) $Insert_a$ (x,y) denotes inserting x right after y; (3) $Insert_b$ (x,y) denotes inserting x just before y. Then, four effective neighborhood structures are given as follows:

NS1: Selecting two operations within a critical block. Inserting one just before the front of the critical block and the other right after the rear of the critical block. The more detail description can refer to: Balas and Vazacopoulos (1998). NS1 can be formulated as follows:

$$NS1 = \{ Insert_b(x, CB_1) \cap Insert_a(y, CB_n) \mid x, y \in \{CB_2, \ldots, CB_{n-1}\} \} \tag{29}$$

NS2: Swapping the first two operations of the critical block with the last two operations of the critical block. The more detail description can refer to: Nowicki and Smutnicki (1996). NS2 can be formulated as follows:

$$NS2 = \{ Swap(CB_1, CB_2) \cap Swap(CB_{n-1}, CB_n) \} \tag{30}$$

NS3: Inserting the front of the critical block into the inner of the critical block, which can be formulated as follows:

$$NS3 = \{ Insert_a(CB_1, x) \mid x \in \{CB_2, \ldots, CB_{n-1}\} \} \tag{31}$$

NS4: Inserting the rear of the critical block into the inner of the critical block, which can be formulated as follows:

$$NS4 = \{ Insert_b(CB_2, x) \mid x, y \in \{CB_2, \ldots, CB_{n-1}\} \} \tag{32}$$

(2) **random search**

The better search based on the critical block is easy to fall into local optimum. Thus, random search is designed to help the algorithm to jump out of the local optimum. One neighborhood structure for random search are given as follows:

NS5: Random search is not performed on the critical block, but random swap operations on grey wolf individual, which can be formulated as follows:

$$NS5 = \{ Swap(x, y) \mid x, y \in X_o \} \tag{33}$$

The neighborhood solutions are generated by hybrid search strategy, and pseudo-code of this process is given in Fig. 12.

5.3.3. Procedure of improved tabu search

The details of steps can be described as follows:

Step 1: Set the parameters of TS, set tabu list empty and let *counter* = 0.
Step 2: Adjust parameters dynamically through dynamic adjustment mechanism.
Step 3: If *counter* < *TS_iter*, perform **Steps 4**; else, output the

procedure: hybrid search strategy
input: the critical operation set, *temp_nb*
output: neighborhood solutions
begin
for $n = 1$ to *temp_nb*
Randomly select critical operations on a machine in M denoted by *curCB*.
if *curCB* contains more than three operations
Apply the NS1, NS2 and NS5 in Eq. (29), (30) and (33) to search the neighboorhood solutions, and the ratios are 0.4: 0.4: 0.2.
else if *curCB* contains more than one operation
Apply the NS3, NS4 and NS5 in Eq. (31), (32) and (33) to search the neighboorhood solutions, and the ratios are 0.4: 0.4: 0.2.
else
Apply the NS5 in Eq. (33) to search the neighboorhood solutions.
end for
end

Fig. 12. Pseudo-code of hybrid search strategy.

archive.
Step 4: Generate neighborhood solutions by hybrid search strategy. Evaluate the neighborhood solutions by fast non-dominated sorting, and update archive by the non-dominated solutions of neighborhood solutions.
Step 5: Select candidate solutions from the non-dominated solutions of neighborhood solutions.
Step 6: If candidate solutions dominate current solution, set the best one as current solution, and perform **Step 8**; else, perform **Step 7**.
Step 7: Select the best solution which is not tabu solution in candidate solutions and set it as current solution.
Step 8: Update the tabu list.

5.4. Global search by social hierarchy and modified search operator

5.4.1. Social hierarchy

In the multi-objective optimization problem, the optimal result is usually a set of non-dominated solutions. It is difficult to directly determine three non-dominant solutions only through Pareto dominance. In this case, the crowding distance is adopted to evaluate the quality of gray wolf individual, which can reflect the degree of crowd of the gray wolf individual in the objective space. The crowding distance D_g of the gray wolf individual g can be calculated as follows:

$$D_g = \begin{cases} \infty, & \text{if } g = 1 \text{ or } g = l \\ |f_1(g+1) - f_1(g-1)| + |f_2(g+1) - f_2(g-1)| & \\ \quad + |f_3(g+1) - f_3(g-1)|, & \text{otherwise} \end{cases} \tag{34}$$

where $g+1$, $g-1$ are gray wolf individuals around gray wolf individual g in the objective space. l is the number of gray wolf individualss. $f_1(g)$, $f_2(g)$ and $f_3(g)$ represent the three objective function values of gray wolf individual g respectively.

In this paper, the α, β and δ are obtained by the following assumptions:

(1) If there are more than two solutions in first front of the archive, three solutions are randomly selected from the first front of the archive. Calculating the crowding distances of three solutions, the biggest one is α, the second biggest one is β and the rest is δ.
(2) If there are only two solutions in first front of the archive, α is the one with a bigger crowding distances and the rest is β. δ is randomly selected from the second front of the archive.
(3) If there are only one solutions in first front of the archive and more than two solutions in the second front of the archive, α is the solution in the first front of the archive. Randomly selecting two solutions from the second front of the archive, β is the one with bigger crowding distances and the rest is δ.
(4) If there are only one solutions in first front of the archive and one solutions in the second front of the archive, α is the solution in the first front of the archive and β is the solution in the second front of the archive. δ is randomly selected from the third front of the archive.

5.4.2. Modified search operator

Due to the fact that GWO is mostly applied to tackle continuous function problems, it may generate infeasible solutions if using search operator of the original GWO to tackle CPSP. To ensure that the search operator is suitable for tackling the CPSP, a modified search operator is designed to update the position of gray wolf individual, which is formulated as follows:

$$\vec{X}_{new}(t) = \begin{cases} Cro(\vec{X}(t), \vec{X}_\alpha(t)) & \text{if rand} < \frac{1}{3} \\ Cro(\vec{X}(t), \vec{X}_\beta(t)) & \text{else if rand} < \frac{2}{3} \\ Cro(\vec{X}(t), \vec{X}_\delta(t)) & \text{otherwise} \end{cases} \tag{35}$$

where function *Cro* ( ) denotes performing crossover, which consists of two parts: X_O and X_m. The POX proposed by Zhang, Rao, and Li (2008) is adopted for X_O, and the RPX proposed by Zhang, Wang, and Xu (2017) is adopted for X_m. The rand is a uniform random value between [0, 1].

In addition, a probability p is set to filter out the bad individuals while retaining good individuals. This operation is defined as follows:

$$\vec{X}(t+1) = \begin{cases} Mut(\vec{X}_{new}(t)) & \text{if } f(\vec{X}(t)) \prec f(\vec{X}_{new}(t)) \text{ and } r < p \\ \vec{X}_{new}(t) & \text{otherwise} \end{cases} \tag{36}$$

where function *Mut* ( ) denotes performing mutation. Two mutation operators are designed in this paper, which are insertion mutation for X_O and random mutation for X_m. The variables r and p are uniform

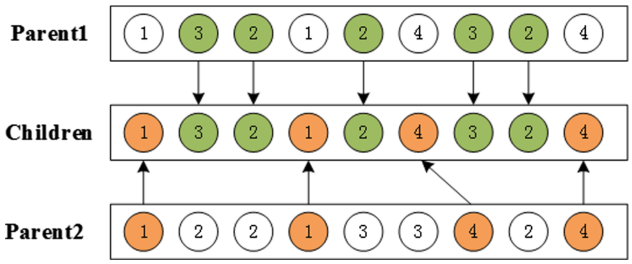

Fig. 13. Example of POX.

random values between [0, 1], symbol ≺ indicates dominance.

POX The example of POX is given in Fig. 13. The X_O of Parent1 and Parent2 are [1,3,2,1,2,4,3,2,4] and [1,2,2,1,3,3,4,2,4]. The job set $O = \{1,2,3,4\}$ is divided into two groups randomly: $O_1 = \{2,3\}$ and $O_2 = \{1,4\}$. The elements in Parent1 which belongs to O_1 are appended to the same position in Children; the elements in Parent2 which belongs to O_2 are appended to the remaining empty positions in Children.

RPX First, a D-dimensional vector R is generated randomly, and the elements in the vector R are uniform random values between [0,1]. The index of the position whose value is less than pf is marked, and the corresponding operations of marked index in Parent1 is recorded. Then, the corresponding elements of recorded operations which belongs to X_m of Parent2 are appended to the position in X_m of Children; the corresponding elements of unrecorded operations which belongs to X_m of Parent1 are appended to the same positions in X_m of Children. pf represents self-adaption probability, which is formulated as follows:

$$pf = pf_{\max} - \frac{pf_{\max} - pf_{\min}}{It} \times MaxIt \tag{37}$$

where pf_{max} and pf_{min} are the maximum and minimum self-adaption probabilities, It is current iteration, $MaxIt$ is maximum iteration. The example of RPX is given in Fig. 14, where the value of pf is 0.4.

6. Experiments and discussion

In this section, a real-world case is studied. To measure the Pareto front (PF) obtained by algorithms, three performance metrics are employed. Then, the algorithm's parameters are tuned by the Taguchi method of design of experiment. In addition, the effectiveness of each improvement strategy of the proposed algorithm is investigate. At the end, the HDMGWO is compared with other multi-objective algorithms to further evaluate the performance of the HDMGWO on CPSP. All algorithms are coded in Matlab R2016a and run on a computer with Intel Core i5, 2.60 GHz, 4 GB RAM, with Windows 7 operating system.

6.1. Case description

Consider the following CPSP from a realistic foundry shop. The foundry shop is required to process 15 castings in a production cycle and there are 26 machines are available for processing. The other relative datas are given in Tables 3–6. The transportation rate is 10 ¥/h in this paper.

6.2. Performance metrics

Three performance metrics are used to measure the PF obtains by these multi-objective algorithms, which are given as follows:

(1) Mean ideal distance (MID). This metric measures the convergence rate of the algorithm. The lower value of MID, the better quality and performance of algorithm.
(2) Spread of non-dominated solution (SNS). This metric measures the diversity of solutions. The higher value of SNS denotes the better diversity of solutions.
(3) Percentage of domination (POD). This metric measures the ability of an algorithm to dominate the solutions of other algorithms. The higher value of POD means that the algorithm is more effective than other algorithms.

The more detail illustrations and formulations of these metrics can refer to Fathollahi-Fard, Hajiaghaei-Keshteli, and Tavakkoli-Moghaddam (2018) and Govindan, Jafarian, Khodaverdi, and Devika (2014).

6.3. Parameter settings

Since the parameters of metaheuristics can significantly affect the performance of metaheuristics, it is necessary to adjust the parameters of metaheuristics (Fathollahi-Fard, Hajiaghaei-Keshteli, & Mirjalili, 2018b; Shao, Pi, & Shao, 2017). The Taguchi method of design of experiment (DOE) (Meng & Pan, 2017) is employed in this paper to determine appropriate parameter values of HDMGWO by using the real-world case. There are three key parameters: the population size $nPop$, the local search times TS_iter and the initialization ratio R. The combinations of different values of these parameters are listed in Table 7.

In order to avoid randomness of the test results, the HDMGWO runs 30 times independently for each parameter combination. The response variable value (RV) is presented in Eq. (38). Two main concepts exist in this response, including convergence and diversity, which are introduced in the prior sub-section. The average response variable (ARV) value during 30 run times is considered. The larger value of ARV is, the better the combination is. According to the number of parameters and factor levels, the orthogonal array L_{16} is selected. Setting the termination time of the algorithm as 500 s, the orthogonal array and obtained ARV values are listed in Table 8. We analyze the orthogonal table through the statistical analysis tool Minitab. The significance rank of each parameter are list in Table 9 and the trend of each factor level is illustrated in Fig. 15.

$$RV = -10 \times \log\left(\frac{MID}{SNS}\right) \tag{38}$$

From Table 9 and Fig. 15, we can see that the local search times TS_iter is the most significant one among the three parameters. A small number of local search times can search the best solution better. The

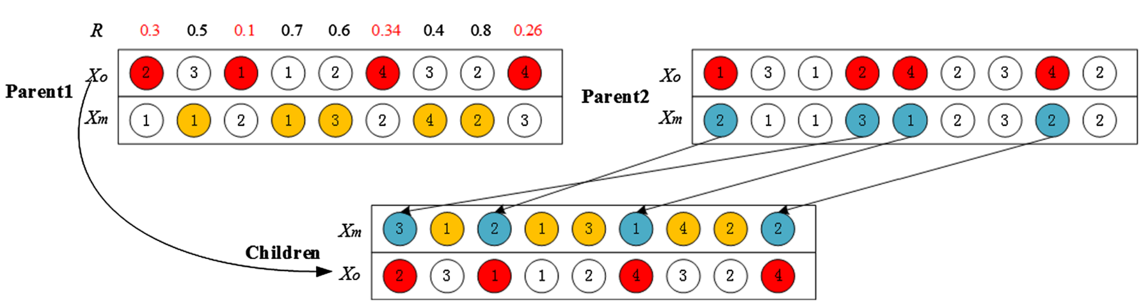

Fig. 14. Example of RPX.

Table 3
The alternative machines and *PI* of the operation.

	Operation										
	Modeling	Melting	Clearing	Degating	Heat treatment	Weld repairing	Throwing pill	Sand blasting	Grinding	Machining	Flaw detection
M	[M_1-M_4]	[M_5-M_7]	[M_8-M_{10}]	[M_{11},M_{12}]	[M_{13},M_{14}]	[M_{15},M_{16}]	[M_{17},M_{18}]	[M_{19},M_{20}]	[M_{21},M_{22}]	[M_{23},M_{24}]	[M_{25},M_{26}]
PI	–	[$20,4^{+1}$]	–	–	[$18,6^{+1}$]	–	–	–	–	–	–

population size *nPop* ranks second. But the difference between the Delta values of *TS_iter* and *nPop* is narror. A large value of *nPop* makes the HDMGWO converge slowly, while a small value may cause premature convergence. As for the initialization ratio *R*, a large value of initialization ratio can generate the better the initial solutions, which is helpful for searching the best solution. According to the analysis of experimental results, it suggests *TS_iter* = 10, *nPop* = 300 and *R* = 0.8 for the following tests and comparisons.

6.4. Performance analysis of each improvement strategy

As stated in Section 5, there are four main improvement procedures of HDMGWO, including the RTP strategy, the improved tabu search, and the improved social hierarchy strategy. We will test the effectiveness of each improvement procedures.

6.4.1. Effectiveness of the RTP strategy

As stated before, a RTP strategy is designed to improve the quality of initial population. In order to investage the effectiveness of RTP strategy, we compared the proposed HDMGWO with its variant with a random initial population which denoted by $\text{HDMGWO}_{\text{rip}}$. The parameters of HDMGWO and $\text{HDMGWO}_{\text{rip}}$ are the same for a fair comparison. Setting the termination time of the two algorithms as 500 s, the evaluation results of three performance metrics during 30 independent runs are listed in Table 10.

It can be observed from Table 10 that the HDMGWO significantly outperforms the $\text{HDMGWO}_{\text{rip}}$ in terms of MID, SNS and POD metrics. As mentioned above, MID is a convergence performance metric, the lower MID of HDMGWO indicates that the RTP strategy makes the HDMGWO have a better convergence toward the possible Pareto optimal solutions. As for SNS, HDMGWO is better than $\text{HDMGWO}_{\text{rip}}$ because HDMGWO can obtain the higher value, which means that the RTP strategy can maintain the diversity of population well. The higher POD of HDMGWO means that the RTP strategy can significantly improve the quality of population.

Table 5
The JTT between two adjacent operations.

Operation	O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8	O_9	O_{10}	O_{11}
O_1	*inf*	$\begin{bmatrix} 2&1&1 \\ 4&3&2 \\ 1&1&2 \\ 2&3&2 \end{bmatrix}$	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*
O_2	*inf*	*inf*	$\begin{bmatrix} 2&4&3 \\ 2&3&2 \\ 3&2&2 \end{bmatrix}$	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*
O_3	*inf*	*inf*	*inf*	0	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*
O_4	*inf*	*inf*	*inf*	*inf*	$\begin{bmatrix} 2&3 \\ 2&1 \end{bmatrix}$	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*
O_5	*inf*	*inf*	*inf*	*inf*	*inf*	$\begin{bmatrix} 3&2 \\ 4&3 \end{bmatrix}$	*inf*	*inf*	*inf*	*inf*	*inf*
O_6	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	0	*inf*	*inf*	*inf*	*inf*
O_7	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	0	*inf*	*inf*	*inf*
O_8	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	0	*inf*	*inf*
O_9	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	$\begin{bmatrix} 3&4 \\ 2&2 \end{bmatrix}$	*inf*
O_{10}	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	$\begin{bmatrix} 1&2 \\ 1&1 \end{bmatrix}$
O_{11}	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*	*inf*

Table 6
The processing rate of machine.

	Machine												
	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}	M_{11}	M_{12}	M_{13}
F_k^v	80	70	75	80	75	70	80	35	40	40	30	35	45
F_k^s	15	15	20	20	20	15	15	15	10	15	5	5	10
	Machine												
	M_{14}	M_{15}	M_{16}	M_{17}	M_{18}	M_{19}	M_{20}	M_{21}	M_{22}	M_{23}	M_{24}	M_{25}	M_{26}
F_k^v	50	30	35	35	40	40	40	10	10	35	30	20	20
F_k^s	10	15	15	10	15	10	10	10	10	10	5	10	10

Table 4
The delivery time, raw material cost and processing time of the job.

Job	Delivery time (h)	Raw material cost (¥)	Processing time (h)										
			O_1	O_2	O_3	O_4	O_5	O_6	O_7	O_8	O_9	O_{10}	O_{11}
Job 1	168	3460	[13,12,14,10]	[1,2,1]	[12,16,10]	[12,10]	[10,12]	[3,4]	–	–	–	–	–
Job 2	240	4560	[19,20,20,24]	[2,4,1]	[12,14,15]	[16,14]	[14,15]	[6,4]	–	–	–	–	–
Job 3	288	3765	[11,10,10,12]	[2,1,1]	[14,16,14]	[10,11]	[10,12]	[4,6]	[8,10]	[6,7]	[6,5]	[13,16]	[5,6]
Job 4	240	6030	[22,24,23,20]	[1,2,1]	[24,26,25]	[12,14]	[12,10]	[10,8]	–	–	–	–	–
Job 5	240	4835	[23,20,24,21]	[1,1,1]	[22,24,20]	[10,9]	[15,19]	[6,5]	–	–	–	–	–
Job 6	288	4860	[24,20,26,23]	[1,1,2]	[24,27,25]	[12,9]	[16,15]	[11,14]	[11,12]	[8,10]	[8,6]	[12,14]	[4,5]
Job 7	360	8430	[20,21,24,22]	[1,1,2]	[24,26,22]	[12,10]	[14,16]	[4,6]	[10,9]	[12,11]	[7,10]	[15,14]	[8,6]
Job 8	360	4300	[22,25,24,20]	[2,2,1]	[20,24,22]	[12,14]	[15,16]	[6,8]	[8,10]	[13,11]	[9,8]	[12,10]	[7,6]
Job 9	504	9070	[50,48,49,52]	[2,3,2]	[38,32,34]	[22,20]	[21,22]	[10,8]	[12,11]	[13,12]	[22,24]	[45,47]	[5,8]
Job 10	504	9240	[48,46,50,44]	[2,4,3]	[32,36,35]	[20,21]	[20,19]	[9,10]	[9,10]	[12,12]	[21,23]	[48,46]	[6,4]
Job 11	240	3750	[20,24,21,22]	[1,1,1]	[24,20,24]	[12,14]	[13,15]	[6,4]	–	–	–	–	–
Job 12	360	4350	[23,22,20,24]	[2,2,1]	[24,25,24]	[12,10]	[15,16]	[10,7]	[8,10]	[7,6]	[10,11]	[18,20]	[3,5]
Job 13	360	5320	[24,20,26,23]	[1,1,2]	[24,22,24]	[12,9]	[16,15]	[11,14]	[10,13]	[8,9]	[11,12]	[17,18]	[9,6]
Job 14	360	6350	[25,24,26,24]	[2,1,1]	[24,23,22]	[11,10]	[15,18]	[8,5]	[10,10]	[7,9]	[7,6]	[12,10]	[4,6]
Job 15	504	9230	[50,48,45,48]	[2,1,2]	[34,30,31]	[19,22]	[20,18]	[12,11]	[12,10]	[14,13]	[20,21]	[46,44]	[8,7]

Table 7
Combinations of parameter values.

Parameters	Factor level			
	1	2	3	4
nPop	100	200	300	400
TS_iter	10	15	20	25
R	0.5	0.6	0.7	0.8

Table 8
Orthogonal array and ARV values.

Experiment number	Factor			ARV
	nPop	*TS_iter*	*R*	
1	1	1	1	15.98
2	1	2	2	15.50
3	1	3	3	16.01
4	1	4	4	15.43
5	2	1	2	16.45
6	2	2	1	15.94
7	2	3	4	15.99
8	2	4	3	15.53
9	3	1	3	17.12
10	3	2	4	16.90
11	3	3	1	16.76
12	3	4	2	16.20
13	4	1	4	17.44
14	4	2	3	16.58
15	4	3	2	16.90
16	4	4	1	15.64

Table 9
Response value.

Level	*nPop*	*TS_iter*	*R*
1	15.73	16.74	16.08
2	15.98	16.23	16.26
3	16.74	16.41	16.31
4	16.64	15.70	16.44
Delta	1.01	1.05	0.36
Rank	2	1	3

6.4.2. Effectiveness of the improved tabu search

To evaluate the effectiveness of the improved tabu search, HDMGWO is compared to its variant with a original tabu search, which is denoted by HDMGWO_{ots}. In HDMGWO_{ots}, the tabu list length is 20, the neighborhood solutions size is 30 and the neighborhood solutions set is obtained by random swapping. The others parameters of HDMGWO and HDMGWO_{ots} are the same for a fair comparison. Setting the termination time of the two algorithms as 500 s, the evaluation results of three performance metrics during 30 independent runs are listed in Table 11.

It is obvious that the HDMGWO has the advantage over the HDMGWO_{ots} because the HDMGWO is significantly better than the HDMGWO_{ots} with regard to the both three metrics. This means that the improved tabu search has better local search ability than original tabu search. The comparison results of the two algorithms are given in Table 12 for further illustration, where Δ_{min}, Δ_{max} and Δ_{avg} represent minimum, maximum and mean in the 30 runs. From Table 12, we can see that the three objectives are improved by using the improved tabu search. The outperformance of HDMGWO can be attributed to the dynamic adjustment mechanism and hybrid search strategy of improved

Table 10
The value of all metrics between HDMGWO and HDMGWO_{rip}.

Algorithm	MID	SNS	POD
HDMGWO	**5482.38**	**250365.10**	**0.75**
HDMGWO_{rip}	12241.61	246891.58	0.25

The best values are shown in bold.

Table 11
The value of all metrics between HDMGWO and $\text{HDMGWO}_{ots.}$

Algorithm	MID	SNS	POD
HDMGWO	**5482.38**	**250365.10**	**0.60**
HDMGWO_{ots}	6490.89	248285.32	0.40

The best values are shown in bold.

Table 12
The comparison results of HDMGWO and $\text{HDMGWO}_{ots.}$

	HDMGWO_{ots}			HDMGWO		
	Δ_{min}	Δ_{max}	Δ_{avg}	Δ_{min}	Δ_{max}	Δ_{avg}
f_1	301	363	325.71	**298**	**340**	**320.37**
f_2	242,295	261,520	250087.16	**241,625**	**255,570**	**248714.58**
f_3	0	78	12.48	**0**	**54**	**8.10**

The best values are shown in bold.

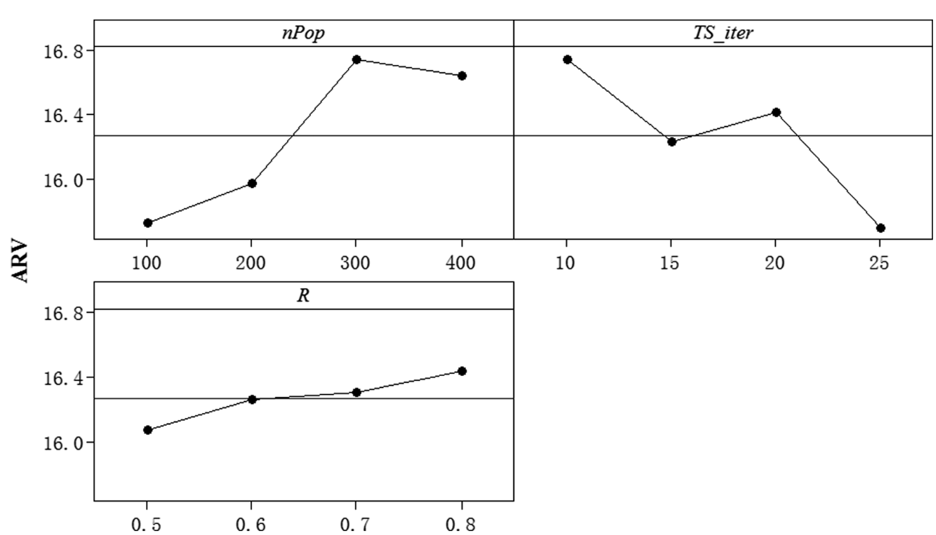

Fig. 15. Fact level trend of HDMGWO.

Table 13
The value of all metrics between HDMGWO and $HDMGWO_{rwm}$.

Algorithm	MID	SNS	POD
HDMGWO	**5482.38**	**250365.10**	**0.5**
$HDMGWO_{rwm}$	6070.26	245844.55	**0.5**

The best values are shown in bold.

tabu search, which improve the searching efficiency and searching quality of tabu search simultaneously.

6.4.3. Effectiveness of the improved social hierarchy strategy

To evaluate the effectiveness of the improved social hierarchy strategy, HDMGWO is compared to its variant in which three best solutions are selected by a roulette-wheel method. This variant is denoted by $HDMGWO_{rwm}$. The parameters of HDMGWO and $HDMGWO_{rwm}$ are the same for a fair comparison. Setting the termination time of the two algorithms as 500 s, the evaluation results of three performance metrics during 30 independent runs are listed in Table 13.

It can be seen from Tables 13 that the performance of the proposed algorithm can be improved by using the improved social hierarchy strategy. The HDMGWO has a good convergence rate since it has a better MID value. As for SNS, the HDMGWO also get higher value than $HDMGWO_{rwm}$, which indicates that the improved social hierarchy strategy has a better exploration ability than roulette-wheel method. The POD values of the two algorithms are equal. Hence, the HDMGWO can keep a good balance between exploration and exploitation by using the improved social hierarchy strategy.

6.5. Comparisons of HDMGWO with other algorithms

To further evaluate the effectiveness of the proposed HDMGWO, we compare it with other multi-objective algorithms, including NSGA-II (Deb et al., 2002), SPEA2 (Zitzler, Laumanns, & Thiele, 2001), MODVOA (Lu, Li, et al., 2017), a multi-objective version of HDPSO (Zhang et al., 2017) and MOGWO. Among them, NSGA-II and SPEA2 are very popular multi-objective evolutionary algorithms, which are usually used as the compared algorithms. MODVOA and HDPSO are two newly reported metaheuristics which have been proven their ability to solve FJSP. MOGWO denotes the multi-objective grey wolf optimizer without any improved strategy, which can further demonstrate the effectiveness of the improvement strategy in HDMGWO. To make a fair comparison between these algorithms, the same encoding scheme and initialization are used as HDMGWO. Furthermore, NSGA-II and SPEA2 use the same crossover operator and mutation operator as HDMGWO. The other parameter settings of those algorithms are the same as those designed in the literature. Each algorithm runs 30 times independently and the termination time of each algorithm is set as 500 s.

The PF obtained by above six algorithms is given in Fig. 16. It can be seen that the HDMGWO can approach closer to the true PF than NSGA-II, SPEA2, MODVOA, HDPSO and MOGWO, which verifies the

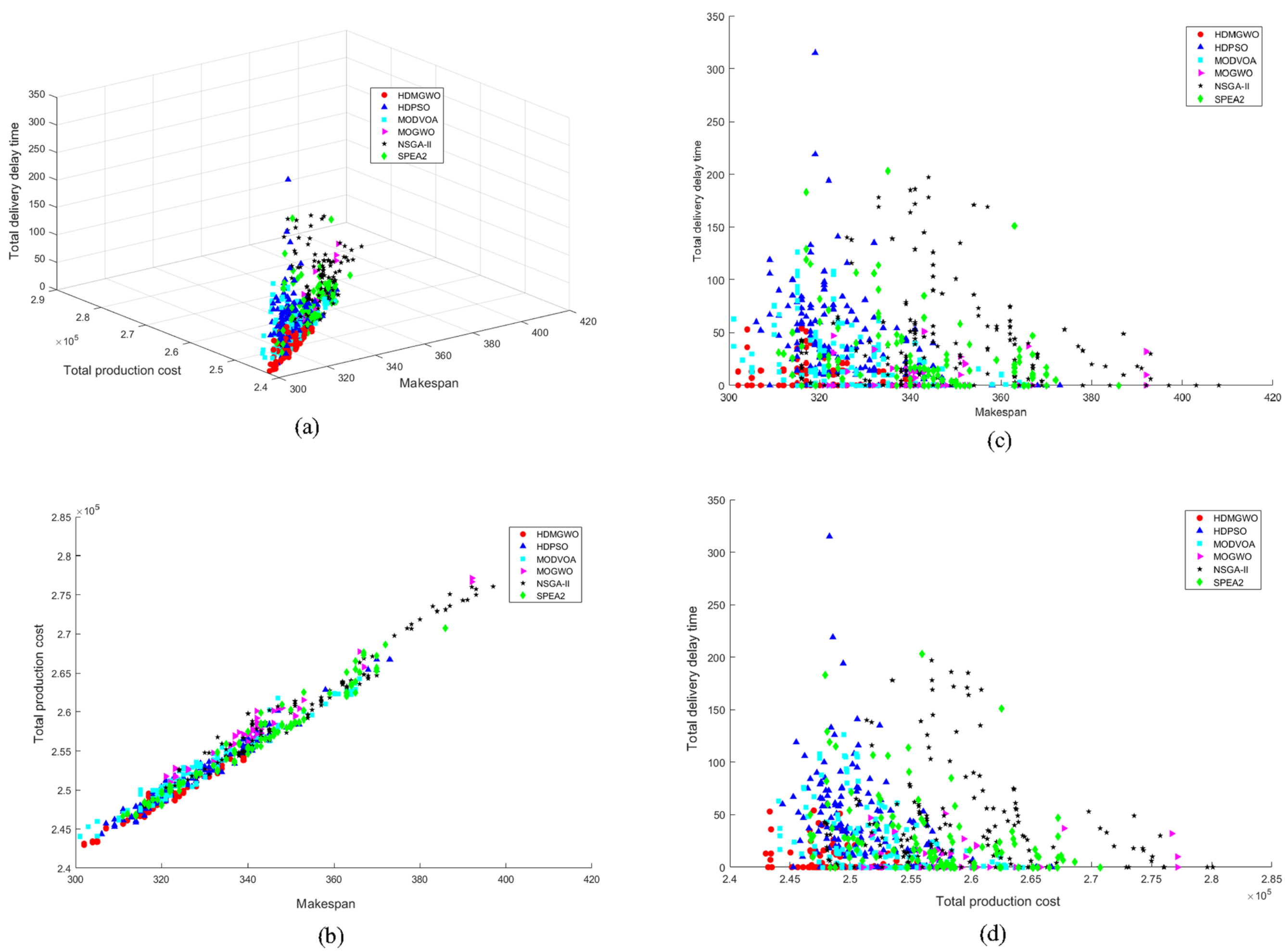

Fig. 16. Pareto fronts obtained by different algorithms under different angles, (a)PF by different algorithms with three criteria, (b) PF with makespan and total production cost, (c) PF with makespan and total delivery delay time, (d) PF with total production cost and total delivery delay time.

Table 14
The value of all metrics obtained by HDMGWO, NSGA-II, SPEA2, MODVOA, HDPSO and MOGWO.

Algorithm	MID	SNS	POD
HDMGWO	**5482.38**	250365.10	**0.27**
NSGA-II	14746.80	249043.26	0.12
SPEA2	12392.06	247901.31	0.08
MODVOA	9128.83	249766.59	0.24
HDPSO	6899.13	**250505.09**	0.19
MOGWO	14232.44	248742.15	0.10

The best values are shown in bold.

effectiveness of the proposed HDMGWO. It also implies that the proposed HDMGWO can obtain the better non-dominated solutions than in terms of quality of solutions. Fig. 16(a) provides a 3-dimensional view with three criteria. From Fig. 16(b), we can see that the point with minimum makespan and minimum total production cost is obtained by the proposed HDMGWO. Fig. 16(c) presents a 2-dimensional view with makespan and total delivery delay time. It is clear that the proposed HDMGWO can obtain the better solutions than other five algorithms. Fig. 16(d) gives a graphical view only considering total production cost and total delivery delay time. It also can be seen the superiority of the proposed HDMGWO. These observations are consistent with our views that the proppDsed HDMGWO can obtain the better non-dominated solutions than NSGA-II, SPEA2, MODVOA, HDPSO and MOGWO.

The performance and efficiency of the above six algorithms are evaluated by performance metrics like MID, SNS, and POD for obtained PF. The results are listed in Table 14. It can be observed that the proposed HDMGWO is significantly better than other five algorithms in terms of MID and POD while the HDPSO is superior to other five algorithms with SNS. The difference between the SNS of HDMGWO and HDPSO is narror. This indicate that the proposed HDMGWO has the competitive performance among these five algorithms in terms of convergence, diversity and the quality of solutions. The reasons for the superiority of HDMGWO can be summarized as follows: First, the RTP strategy improve the quality of initial solutions which facilitates improved tabu search for local search. Second, to enhance exploitation, the improved tabu search with the dynamic adjustment mechanism and hybrid search strategy is designed which make the HDMGWO quickly converge to the optimal solutions. Third, to enhance exploration, the population update their position with modified search operator based on the guidence of the three best solutions selected by improved social hierarchy. The improved social hierarchy maintains the diversity of population and the modified search operator explores new unvisited areas of the search space.

To investigate the best algorithm among above six algorithms for solving the case of the real foundry enterprise, the Boxplot of three performance metrics obtained by six algorithms during 30 runs are shown in Fig. 17. This indicates the superiority and the performance of the proposed HDMGWO in this paper.

To compare the quality of obtained solution sets of algorithms visually, the parallel coordinates plot (Li, Zhen, & Yao, 2017) is adopted. Since the large difference in range of three objective function, the normalized values of solutions are considered in this paper. The normalized value (*NV*) can be formulated as follows:

$$NV = \frac{S_c - S_b}{S_w - S_b} \tag{39}$$

where S_c is the current value of algorithm, S_b is the best value ever found in the problem and S_w is the worse value ever found in the problem. It should be mentioned that the lower value for *NV* is preferred.

From Fig. 18, we can see that the normalized values of three objection functions obtained by the HDMGWO is much better than other five algorithms. Moreover, the normalized values obtained by the HDMGWO have a small range. This indicates that the HDMGWO performs better both in solution quality and stability than other five algorithms.

7. The realistic application of HDMGWO

The proposed HDMGWO has been successfully applied to the intelligent scheduling module of a casting ERP system. The scheduling scheme generated by intelligent scheduling module can give managers effective guidance to develop production plans. In casting production, the design of production plans is often random and empirical, which may cause the waste of resources and cannot get the highest production efficiency. But our proposed HDMGWO can provide managers with a set of effective scheduling schemes to improve the production cycle, rate of timely delivery and production cost. Managers can select the most suitable one from the non-dominated solutions to satisfy different demands. In Fig. 19, the 15 jobs in the case study are selected to generate a scheduling scheme (see Fig. 20) in the intelligent scheduling module. The fitness values of this scheduling scheme are [305,243795,0]. This scheduling scheme ensures that all jobs can be completed before delivery while reduces the production cycle by 18%,

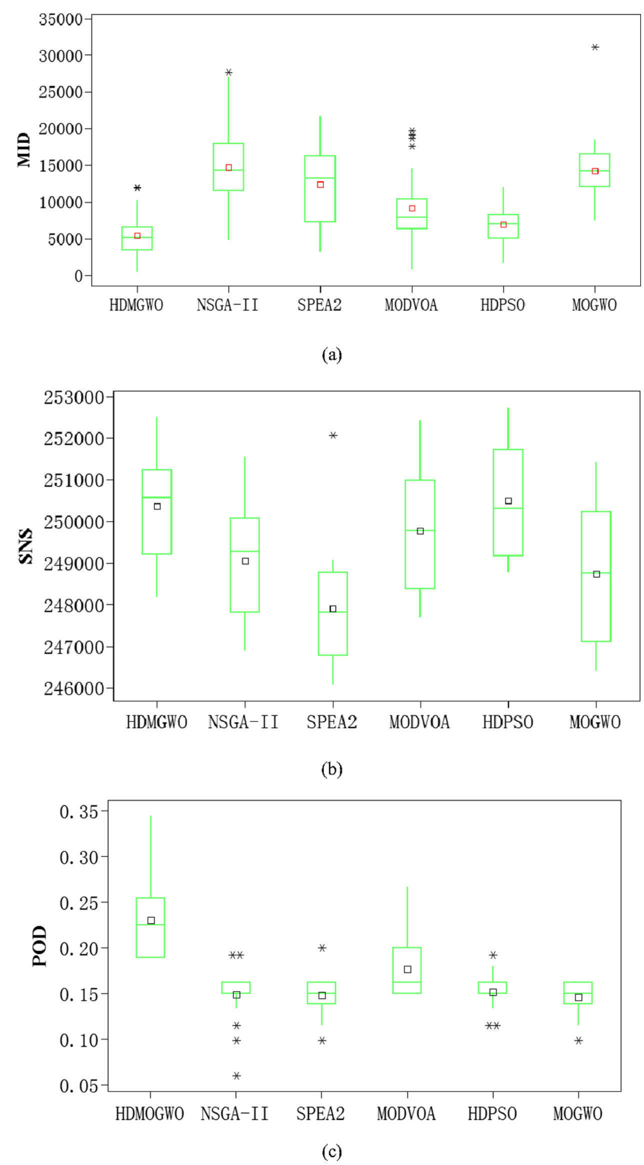

Fig. 17. Boxplot of three performance metrics obtained by different algorithms, (a) Boxplot of MID, (b) Boxplot of SNS, (C) Boxplot of POD.

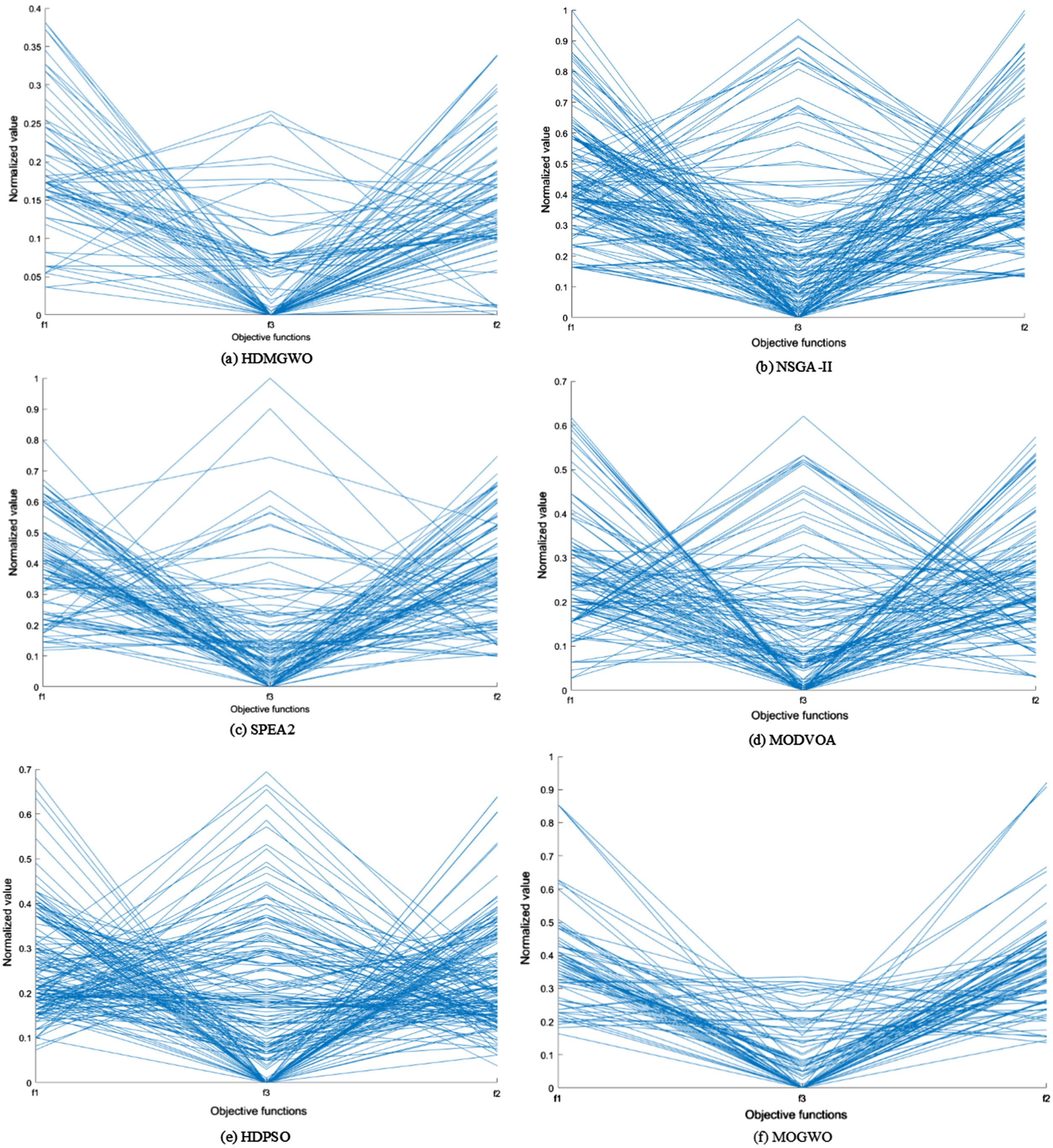

Fig. 18. The parallel coordinates plot for different algorithms.

which is what the managers attach great importance to.

The process that the scheduling scheme drives practical production contains two layers: execution layer and monitoring layer. In execution layer, each foundry shop starts to process jobs according to this scheduling scheme. Since the ERP system need to track the processing status of each job, the workers need transfer related information of the job to the ERP system by scaning barcodes with handheld devices (see Fig. 21). In monitoring layer, the processing information of the job is displayed in the electronic display board.

In realistic application, we also find some problems of the HDMGWO. There are some unpredictable events occur in casting production, like new job insertion and machine breakdown, which may cause the deterioration of the original scheduling scheme. At this stage, our proposed HDMGWO cannot solve those dynamic events well, which is also our motivation to further optimize the proposed HDMGWO in the future work.

8. Conclusions and future work

In this paper, we take the real-world foundry as the research object and extract two constraints that exist in the foundry production, including processing interval constraint and job transportation time. Then, a three-objective mathematical model which considers above two realistic constraints is formulated. The objectives of this model are to minimize makespan, the total production cost and the total delivery delay time simultaneously. To solve this model, a hybrid discrete multi-objective grey wolf optimizer is developed, namely HDMGWO.

With regard to the experimental studies, the DOE is employed to

Fig. 19. The intelligent scheduling module of a casting ERP system.

Fig. 20. The Gantt chart after scheduling.

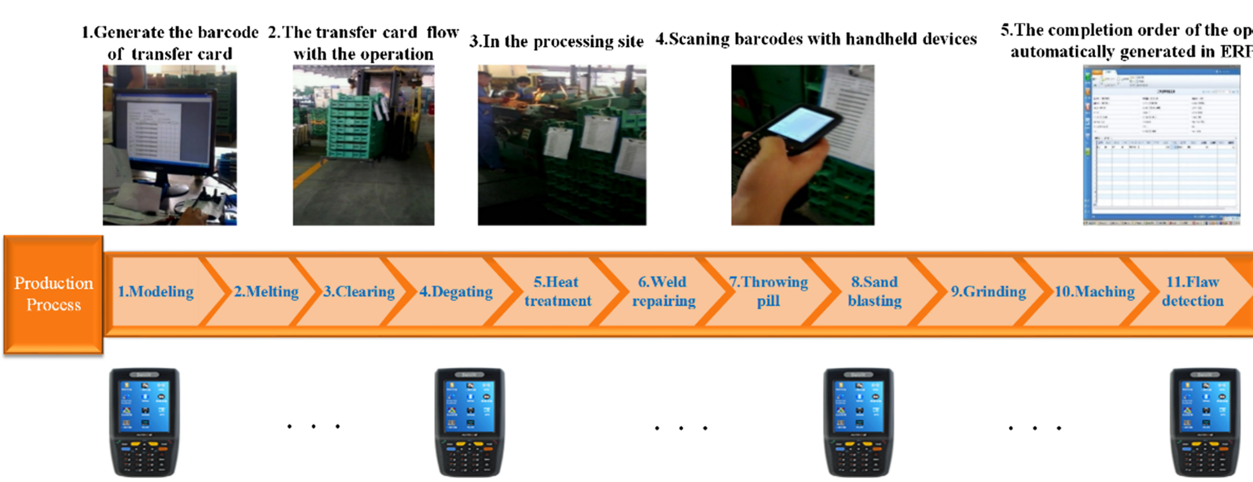

Fig. 21. Transfer information to the ERP system by scaning barcodes.

adjust the parameters of the proposed HDMGWO. Then, we investage the effectiveness of each improvement strategy of the HDMGWO. Besides, the proposed HDMGWO is compared with five multi-objective algorithms including NSGA-II, SPEA2, MODVOA, HDPSO and MOGWO on a real-world case in terms of three evaluation metrics like MID, SNS and POD. The results indicate that the proposed HDMGWO outperforms the other five multi-objective algorithms in terms of the quality of solutions. Furthermore, the real running in a casting ERP system is given to verify the applicability of the proposed scheduling model and the HDMGWO.

Considering the casting production with dynamic feature, such as new job insertion and machine breakdown, our future work is to apply the proposed HDMGWO to the field of dynamic scheduling. Since green manufacturing is increasingly valued by enterprise, we will take account of carbon emissions and energy consumption as the optimization objectives of the algorithm.

Conflict of interest

There is no conflict of interest exits in the submission of this manuscript, and manuscript is approved by all authors for publication.

Acknowledgement

This research work is partially supported by the National Natural Science Foundation of China under Grant No. 51705384.

Appendix A. Supplementary material

Supplementary data to this article can be found online at https://doi.org/10.1016/j.cie.2018.12.061.

References

Balas, E., & Vazacopoulos, A. (1998). Guided local search with shifting bottleneck for job shop scheduling. *Management Science, 44*(2), 262–275.

Blazewicz, J., Domschke, W., & Pesch, E. (1996). The job shop scheduling problem: Conventional and new solution techniques. *European Journal of Operational Research, 93*(1), 2.

Chaudhry, I. A., & Khan, A. A. (2016). A research survey: Review of flexible job shop scheduling techniques. *International Transactions in Operational Research, 23*(3), 551–591.

Chen, A. L., Yang, G. K., & Wu, Z. M. (2008). Production scheduling optimization algorithm for the hot rolling processes. *International Journal of Production Research, 46*(7), 1955–1973.

Chung, T., Sun, H., & Liao, C. (2017). Two new approaches for a two-stage hybrid flowshop problem with a single batch processing machine under waiting time constraint. *Computers & Industrial Engineering, 113*, 859–870.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation, 6*(2), 182–197.

Faris, H., Aljarah, I., Al-Betar, M. A., & Mirjalili, S. (2018). Grey wolf optimizer: A review of recent variants and applications. *Neural Computing and Applications, 30*(2), 413–435.

Fathollahi Fard, A. M., & Hajiaghaei-Keshteli, M. (2018). A bi-objective partial interdiction problem considering different defensive systems with capacity expansion of facilities under imminent attacks. *Applied Soft Computing, 68*, 343–359.

Fathollahi-Fard, A. M., & Hajiaghaei-Keshteli, M. (2018). A stochastic multi-objective model for a closed-loop supply chain with environmental considerations. *Applied Soft Computing, 69*, 232–249.

Fathollahi-Fard, A. M., Hajiaghaei-Keshteli, M., & Mirjalili, S. (2018a). Hybrid optimizers to solve a tri-level programming model for a tire closed-loop supply chain network design problem. *Applied Soft Computing, 70*, 701–722.

Fathollahi-Fard, A. M., Hajiaghaei-Keshteli, M., & Mirjalili, S. (2018b). Multi-objective stochastic closed-loop supply chain network design with social considerations. *Applied Soft Computing Journal, 71*, 505–525.

Fathollahi-Fard, A. M., Hajiaghaei-Keshteli, M., & Tavakkoli-Moghaddam, R. (2018). A bi-objective green home health care routing problem. *Journal of Cleaner Production, 200*, 423–443.

Gao, K. Z., Suganthan, P. N., Pan, Q. K., Chua, T. J., Cai, T. X., & Chong, C. S. (2014). Pareto-based grouping discrete harmony search algorithm for multi-objective flexible job shop scheduling. *Information Sciences, 289*(1), 76–90.

Gen, M., Zhang, W., Lin, L., & Yun, Y. (2017). Recent advances in hybrid evolutionary algorithms for multiobjective manufacturing scheduling. *Computers & Industrial Engineering, 112*, 616–633.

Gendreau, M., Iori, M., Laporte, G., & Martello, S. (2006). A Tabu search algorithm for a routing and container loading problem. *Transportation Science, 40*(3), 342–350.

Glover, F. (1990). Tabu search: A tutorial. *Interfaces, 20*(4), 74–94.

Gong, X., Deng, Q., Gong, G., Liu, W., & Ren, Q. (2018). A memetic algorithm for multi-objective flexible job-shop problem with worker flexibility. *International Journal of Production Research, 56*(7), 2506–2522.

Govindan, K., Jafarian, A., Khodaverdi, R., & Devika, K. (2014). Two-echelon multiple-vehicle location–routing problem with time windows for optimization of sustainable supply chain network of perishable food. *International Journal of Production Economics, 152*, 9–28.

Hajiaghaei-Keshteli, M., & Fathollahi-Fard, A. M. (2018). A set of efficient heuristics and metaheuristics to solve a two-stage stochastic bi-level decision-making model for the distribution network problem. *Computers & Industrial Engineering, 123*, 378–395.

Jia, S. J., Yi, J., Yang, G. K., Du, B., & Zhu, J. (2013). A multi-objective optimisation algorithm for the hot rolling batch scheduling problem. *International Journal of Production Research, 51*(3), 667–681.

Jiang, S., Zheng, Z., & Liu, M. (2018). A preference-inspired multi-objective soft scheduling algorithm for the practical steelmaking-continuous casting production. *Computers & Industrial Engineering, 115*, 582–594.

Kato, E. R. R., Aranha, G. D. D. A., & Tsunaki, R. H. (2018). A new approach to solve the flexible job shop problem based on a hybrid particle swarm optimization and Random-Restart Hill Climbing. *Computers & Industrial Engineering, 125*, 178–189.

Khoukhi, F. E., Boukachour, J., & Alaoui, A. E. H. (2017). The "Dual-Ants Colony": A novel hybrid approach for the flexible job shop scheduling problem with preventive maintenance. *Computers & Industrial Engineering, 106*, 236.

Kiani, M., & Yildiz, A. R. (2016). A comparative study of non-traditional methods for vehicle crashworthiness and NVH optimization. *Archives of Computational Methods in Engineering, 23*(4), 723–734.

Komaki, G. M., & Kayvanfar, V. (2015). Grey Wolf Optimizer algorithm for the two-stage assembly flow shop scheduling problem with release time. *Journal of Computational Science, 8*, 109–120.

Kurdi, M. (2017). An improved island model memetic algorithm with a new cooperation phase for multi-objective job shop scheduling problem. *Computers & Industrial Engineering, 111*, 183–201.

Li, X., & Gao, L. (2016). An effective hybrid genetic algorithm and tabu search for flexible job shop scheduling problem. *International Journal of Production Economics, 174*, 93–110.

Li, J., Pan, Q., & Liang, Y. (2010). An effective hybrid tabu search algorithm for multi-objective flexible job-shop scheduling problems. *Computers & Industrial Engineering, 59*(4), 647–662.

Li, J., Pan, Q., Mao, K., & Suganthan, P. N. (2014). Solving the steelmaking casting problem using an effective fruit fly optimisation algorithm. *Knowledge-Based Systems, 72*, 28–36.

Li, L., Sun, L., Kang, W., Guo, J., Han, C., & Li, S. (2016). Fuzzy multilevel image thresholding based on modified discrete grey wolf optimizer and local information aggregation. *IEEE Access, 4*, 6438–6450.

Li, M., Zhen, L., & Yao, X. (2017). How to read many-objective solution sets in parallel coordinates [educational forum]. *IEEE Computational Intelligence Magazine, 12*(4), 88–100.

Liu, Y., Yi, Z., Wu, H., Ye, M., & Chen, K. (2008). A tabu search approach for the minimum sum-of-squares clustering problem. *Information Sciences, 178*(12), 2680–2704.

Lu, C., Gao, L., Li, X., & Xiao, S. (2017). A hybrid multi-objective grey wolf optimizer for dynamic scheduling in a real-world welding industry. *Engineering Applications of Artificial Intelligence, 57*, 61–79.

Lu, C., Li, X., Gao, L., Liao, W., & Yi, J. (2017). An effective multi-objective discrete virus optimization algorithm for flexible job-shop scheduling problem with controllable processing times. *Computers & Industrial Engineering, 104*, 156–174.

Lu, C., Xiao, S., Li, X., & Gao, L. (2016). An effective multi-objective discrete grey wolf optimizer for a real-world scheduling problem in welding production. *Advances in Engineering Software, 99*, 161–176.

Meng, T., & Pan, Q. (2017). An improved fruit fly optimization algorithm for solving the multidimensional knapsack problem. *Applied Soft Computing, 50*, 79–93.

Meng, T., Pan, Q., & Sang, H. (2018). A hybrid artificial bee colony algorithm for a flexible job shop scheduling problem with overlapping in operations. *International Journal of Production Research, 56*(16), 5278–5292.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software, 69*, 46–61.

Nouiri, M., Bekrar, A., Jemai, A., Niar, S., & Ammari, A. C. (2018). An effective and distributed particle swarm optimization algorithm for flexible job-shop scheduling problem. *Journal of Intelligent Manufacturing, 29*(3), 603–615.

Nouiri, M., Bekrar, A., Jemai, A., Trentesaux, D., Ammari, A. C., & Niar, S. (2017). Two stage particle swarm optimization to solve the flexible job shop predictive scheduling problem considering possible machine breakdowns. *Computers & Industrial Engineering, 112*, 595–606.

Nowicki, E., & Smutnicki, C. (1996). A fast Taboo search algorithm for the job shop problem. *Manege Sci, 42*(6), 797–813.

Nuaekaew, K., Artrit, P., Pholdee, N., & Bureerat, S. (2017). Optimal reactive power dispatch problem using a two-archive multi-objective grey wolf optimizer. *Expert Systems with Applications, 87*, 79–89.

Pacciarelli, D., & Pranzo, M. (2004). Production scheduling in a steelmaking-continuous casting plant. *Computers & Chemical Engineering, 28*(12), 2823–2835.

Pholdee, N., Bureerat, S., & Yıldız, A. R. (2017). Hybrid real-code population-based incremental learning and differential evolution for many-objective optimisation of an automotive floor-frame. *International Journal of Vehicle Design, 73*(1–3), 20–53.

Sha, D. Y., & Hsu, C. (2006). A hybrid particle swarm optimization for job shop scheduling problem. *Computers & Industrial Engineering, 51*(4), 791–808.

Shao, Z., Pi, D., & Shao, W. (2017). Self-adaptive discrete invasive weed optimization for the blocking flow-shop scheduling problem to minimize total tardiness. *Computers & Industrial Engineering, 111*, 331–351.

Tang, L., Zhao, Y., & Liu, J. (2014). An improved differential evolution algorithm for practical dynamic scheduling in steelmaking-continuous casting production. *IEEE Transactions on Evolutionary Computation, 18*(2), 209–225.

van Laarhoven, P. J. M., Aarts, E. H. J., & Lenstra, J. K. (1992). Job shop scheduling by simulated annealing. *Operations Research, 40*(1), 113–125.

Wang, L., Zhou, G., Xu, Y., & Liu, M. (2012). An enhanced Pareto-based artificial bee colony algorithm for the multi-objective flexible job-shop scheduling. *The International Journal of Advanced Manufacturing Technology, 60*(9–12), 1111–1123.

Yildiz, A. R. (2012). A comparative study of population-based optimization algorithms for turning operations. *Information Sciences, 210*, 81–88.

Yildiz, A. R. (2013a). Comparison of evolutionary-based optimization algorithms for structural design optimization. *Engineering Applications of Artificial Intelligence, 26*(1), 327–333.

Yildiz, A. R. (2013b). Cuckoo search algorithm for the selection of optimal machining parameters in milling operations. *The International Journal of Advanced Manufacturing Technology, 64*(1–4), 55–61.

Yıldız, B. S. (2017). A comparative investigation of eight recent population-based optimisation algorithms for mechanical and structural design problems. *International Journal of Vehicle Design, 73*(1–3), 208–218.

Yıldız, B. S., & Lekesiz, H. (2017). Fatigue-based structural optimisation of vehicle components. *International Journal of Vehicle Design, 73*(1–3), 54–62.

Yildiz, A. R., & Saitou, K. (2011). Topology synthesis of multicomponent structural assemblies in continuum domains. *Journal of Mechanical Design, 133*(1), 11008.

Yuan, Y., & Xu, H. (2015). Multiobjective flexible job shop scheduling using memetic algorithms. *IEEE Transactions on Automation Science and Engineering, 12*(1), 336–353.

Zhang, L., Gao, L., & Li, X. (2013). A hybrid genetic algorithm and tabu search for a multi-objective dynamic job shop scheduling problem. *International Journal of Production Research, 51*(12), 3516–3531.

Zhang, G., Gao, L., & Shi, Y. (2011). An effective genetic algorithm for the flexible job-shop scheduling problem. *Expert Systems with Applications, 38*(4), 3563–3573.

Zhang, C., Rao, Y., & Li, P. (2008). An effective hybrid genetic algorithm for the job shop scheduling problem. *The International Journal of Advanced Manufacturing Technology, 39*(9), 965.

Zhang, Y., Wang, J., & Liu, Y. (2017). Game theory based real-time multi-objective flexible job shop scheduling considering environmental impact. *Journal of Cleaner Production, 167*, 665–679.

Zhang, J., Wang, W., & Xu, X. (2017). A hybrid discrete particle swarm optimization for dual-resource constrained job shop scheduling with resource flexibility. *Journal of Intelligent Manufacturing, 28*(8), 1961–1972.

Zitzler, E., Laumanns, M. & Thiele, L. (2001). In Evolutionary methods for design, optimization and control with applications to industrial problems. Proceedings of the Eurogen'2001, Athens, Greece, September.