Expert Systems With Applications 126 (2019) 321–339

Contents lists available at ScienceDirect

Expert Systems With Applications

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

A hybrid biogeography-based optimization with variable neighborhood search mechanism for no-wait flow shop scheduling problem

Fuqing Zhao[a,*], Shuo Qin[a], Yi Zhang[b], Weimin Ma[c], Chuck Zhang[d], Houbin Song[a]

[a] School of Computer and Communication Technology, Lanzhou University of Technology, Lanzhou 730050, China
[b] School of Mechanical Engineering, Xijin University Xi'an 710123, China
[c] School of Economics and Management, Tongji University, Shanghai 200092, China
[d] H. Milton Stewart School of Industrial & Systems Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA

ARTICLE INFO

Article history:
Received 28 September 2018
Revised 11 December 2018
Accepted 17 February 2019
Available online 19 February 2019

Keywords:
Biogeography-based optimization
No-wait flow shop scheduling problem
Block neighborhood structure
Markov model
Path relink technique

ABSTRACT

The no-wait flow shop scheduling problem (NWFSP) plays an essential role in the manufacturing industry. Inspired by the overall process of biogeography theory, the standard biogeography-based optimization (BBO) was constructed with migration and mutation operators. In this paper, a hybrid biogeography-based optimization with variable neighborhood search (HBV) is implemented for solving the NWFSP with the makespan criterion. The modified NEH and the nearest neighbor mechanism are employed to generate a potential initial population. A hybrid migration operator, which combines the path relink technique and the block-based self-improvement strategy, is designed to accelerate the convergence speed of HBV. The iterated greedy (IG) algorithm is introduced into the mutation operator to obtain a promising solution in exploitation phase. A variable neighbor search strategy, which is based on the block neighborhood structure and the insert neighborhood structure, is designed to perform the local search around the current best solution in each generation. Furthermore, the global convergence performance of the HBV is analyzed with the Markov model. The computational results and comparisons with other state-of-art algorithms based on Taillard and VRF benchmark show that the efficiency and performance of HBV for solving NWFSP.

1. Introduction

Shop scheduling plays an important role in manufacturing scheduling since an excellent scheduling planning improves the productivity of the company effectively. The shop scheduling problem is classified into single machine scheduling problem with single processor, single machine scheduling problem with parallel processors, flow shop scheduling problem, job shop scheduling problem (Van Laarhoven, Aarts, & Lenstra, 1992). The flow shop scheduling problem (FSP) seems to be a common problem in the field of industrial production. The FSP is divided into the following categories: permutation FSP (PFSP), no-wait FSP (NWFSP), blocking FSP (BFSP) (Riahi, Khorramizadeh, Newton, & Sattar, 2017), no-idle FSP (NIFSP) (Shao, Pi, & Shao, 2018), non-smooth FSP (Ferrer, Guimarans, Ramalhinho, & Juan, 2016) and hybrid FSP (HFSP) (Lei, Liang, & Zheng, 2018), etc.

The NWFSP, which is an extension of the FSP, has been widely applied in various industries, such as the electronics, metals, chemicals and food-processing industries. As the technological reasons, n jobs are processed in the same order on m machines without waiting time between consecutive operations until the whole process is done in NWFSP. Therefore, the starting time of a job on a machine have to be delayed to satisfy the no-wait constraints. The NWFSP with three or more machines was proved by Garey and Johnson (1979) to be one of the strongly NP-hard problems. It is difficult to solve the problems by branch-and-bound or mixed integer programming methods as the problem size increases. It is necessary to try more efficient algorithms for solving the NWFSP.

Evolutionary algorithms (EAs) have gained wide popularity as they solve the complex optimization problems without any gradient information. Various EAs and variants of classical EAs, such as particle swarm optimization (Clerc & Kennedy, 2002), harmony search (Geem, Kim, & Loganathan, 2001), water wave optimization (Zheng, 2015), gravitational search algorithm (Rashedi, Nezamabadi-Pour, & Saryazdi, 2009), have been proposed. In recent years, various heuristics are proposed to solve the NWFSP since they in general obtain high-quality solutions within a reasonable

* Corresponding author.
E-mail addresses: Fzhao2000@hotmail.com (F. Zhao), qinshuo0234@163.com (S. Qin), 1049440546@qq.com (Y. Zhang), mawm@tongji.edu.cn (W. Ma), chuck.zhang@isye.gatech.edu (C. Zhang), 523712418@qq.com (H. Song).

https://doi.org/10.1016/j.eswa.2019.02.023
0957-4174/

time limit. The heuristic algorithms are classified into two main categories: constructive heuristics and meta-heuristics.

Inspired by the migration of species in natural biogeography, the biogeography-based optimization (BBO) was proposed by Simon (2008). In the last decade, the BBO algorithm and its variants have been successfully employed to solve complex problems. A hybrid algorithm which comprises PSO and BBO was presented by Yogesh et al. (2017) for emotion and stress recognition from the speech signal. A learning mechanism for radial basis function networks, which is based on biogeography-based optimization with population competition (BBOPC) algorithm, was proposed by Aljarah, Faris, Mirjalili, and Al-Madi (2018) to improve the economics of electric power planning. A two-stage differential biogeography-based optimization (TDBBO) algorithm, which combines the Gaussian mutation operator, the two-stage migration model and BBO/current-to-select/1, was proposed by Zhao et al. (2019) for solving the global continuous optimization problem. The convergence performance of TDBBO was analyzed with Markov model.

The BBO algorithm and the variants of BBO have been widely applied to scheduling problems. An HBBO algorithm, which is combined with the chaos theory and BBO, was proposed by Wang and Duan (2014) to solve the job shop scheduling problem. The BBO algorithm was combined with some heuristics by Lin (2015) to construct a hybrid algorithm for solving the fuzzy flexible JSP. The BBO was also developed by Rabiee, Jolai, Asefi, Fattahi, and Lim (2016) to address the realistic no-wait hybrid flow shop (NWHFSP) with unrelated parallel machines to minimize mean tardiness. The hybrid discrete biogeography-based (HDBBO) was proposed by Lin (2016) for solving the permutation FSP. For solving the blocking flow shop scheduling problem (BFSP) with maximum completion time criterion, an improved biogeography-based optimization (BBO) was proposed by Liu, Wang, and Zhang (2018).

However, although the BBO algorithm and its variants were widely applied to solve the combinational optimization problems, little attention has been paid to the NWFSP. In this paper, a hybrid biogeography-based optimization with variable neighborhood search is proposed to be an alternative scheme for solving NWFSP. The experimental results based on Reeve's, Taillard's and VRF benchmark show the effectiveness and robustness of HBV. The contributions of this paper are summarized as follows.

- The standard framework of BBO is retained. The NN+MNEH is introduced to init potential population. The iterated greedy mechanism is employed by the mutation operator to improve the exploitation ability of HBV.
- The path relink technique is embedded into the migration operator to improve the exploration ability of HBV. The self-improvement strategy which is based on the block neighborhood structure is designed to improve the quality of solution.
- A modified variable neighborhood search, which is based on the block neighborhood structure and the insert neighborhood structure, is designed to search around the current best solutions in each generation.
- The convergence performance of HBV is analyzed with the Markov model. The convergence process of the candidate solutions is mapped to the state transition process in the Markov chain.

The remainder of this paper is organized as follows: Section 2 describes the background. Section 3 provides the description of NWFSP. A brief description of the standard BBO algorithm is given in Section 4. In Section 5, the proposed HBV algorithm is introduced in detail for NWFSP. In Section 6, the computational evaluation is provided. Section 7 summarizes the conclusion and future work.

2. Background

The heuristic algorithms, which are available for solving the NWFSP, are classified into two main categories, constructive heuristics and meta-heuristics. Several noteworthy constructive heuristics have been proposed for solving the NWFSP over the last few decades. A slope matching (S/M) method, which employs geometrical relationships between the cumulative process times, was designed by Bonney and Gundry (1976). The RAJ, which is based on heuristic preference relations and job insertion, was presented by Rajendran (1994). The average departure time (ADT) heuristic was proposed by Ye, Wei, and Miao (2016). Based on the computational experiment with a large number of instances of various sizes, the ADT performed than compared algorithms. An average idle time (AIT) heuristic was proposed by Ye, Li, and Abedini (2017) to minimize makespan in NWFSP. Compared with the existing heuristics, the AIT achieved the smaller deviations in the same computational complexity. In addition, several other constructive heuristics for the regular flowshop were used to solve the NWFSP, such as the NEH which was proposed by Nawaz, EEE, and Ham (1983).

Various remarkable meta-heuristics have also been developed for solving the NWFSP. The discrete particle swarm optimization (DPSO) was proposed by Pan, Tasgetiren, and Liang (2008a), Pan, Wang, and Zhao (2008b) for solving the NWFSP with both makespan and total flowtime criteria. Several speed-up methods were proposed for the swap and insert neighborhood structure. The improved iterated greedy algorithm (IIGA) was proposed by Pan et al. (2008a,b) for solving the NWFSP. Three speed-up methods, which includes the insertion of a new job into a partial sequence, the insertion and exchange neighborhood moves, were designed by Wang, Li, and Wang (2010) to reduce the time complexity of evaluating the candidate. A composite heuristic which consists of the standard deviation heuristic with double-job insert operator and the iteration operator was proposed by Gao, Xie, Hua, and Li (2012) to improve the quality of solutions. An improved NEH heuristic was presented to construct the initial population. A local search algorithm was also proposed to perform exploitation. A tabu-mechanism improved iterated greedy (TMIIG) was proposed by Ding et al. (2015a,b). In TMIIG, the tabu-based construction strategy is employed to enhance the exploration ability and several neighborhood structures are applied to improve the quality of solutions. In the same year, the block-shifting simulated annealing (BSA) was introduced by Ding et al. (2015a,b). In BSA, the block neighborhood structure was embedded into the framework of simulated annealing. The NWFSP was formulated as the ATSP by Lin and Ying (2016). Two meta-heuristics were proposed to solve the NWFSP problem. Later on, an extended framework of meta-heuristic based on teaching-learning process was developed by Shao, Pi, and Shao (2017a,b). The DWWO algorithm was introduced by Zhao, Liu, Zhang, Ma, and Zhang (2018) for solving the NWFSP. In DWWO, the operators of the original water wave algorithm were redefined to adapt to the NWFSP. The computational results demonstrate that the DWWO outperforms other compared algorithms. A flower pollination algorithm, which is based on the hormone modulation mechanism, was proposed by Qu, Fu, Yi, and Tan (2018) for solving the NWFSP.

In the studies mentioned above, various algorithms were proposed to solve the NWFSP. However, the existing algorithms still struggle with large size problems. One key reason is that the framework of algorithms and the neighborhood structure are incomplete. Variable neighborhood search (VNS) is a popular meta-heuristic proposed by Hansen and Mladenović (1997) systematically exploiting the idea of neighborhood change. In the DPSO which was proposed by Quan Ke Pan et al. (2008a,b) to solve the NWFSP, a variable neighborhood descent (VND) method was

employed to perform the exploitation. An approach based on variable greedy algorithm was proposed by Framinan and Leisten (2008) for solving the permutation flow shop problem with the objective of minimizing the total tardiness. In this study, the VNS concept of varying the neighborhood has been applied to the destruction and construction phases of iterated greed (IG) algorithm. The refreshing VNS (RVNS) was designed by Perez-Gonzalez and Framinan (2010) to solve the due date setting problem in a constrained flow shop with the objective of minimizing the makespan of the new jobs. In the RVNS, the strict sequence which includes the feasible solutions and the relaxed sequence which consists of either feasible or unfeasible solutions are considered. An efficient variable neighborhood search for the flow shop with many batch processing machines (many-BPM) was proposed by Lei and Guo (2011) to minimize total tardiness, maximum tardiness and number of tardy jobs, respectively. In this VNS, an insertion operation and two swap operation are applied to improve the initial permutation. The computational results showed the excellent performance of the VNS on many-BPM flow shop problem. The differential evolution algorithm based on variable neighborhood search (VNSDE) was presented by Zhou (2012) for solving the flow shop problem with the makespan criterion. In VNSDE, the VNS is performed after the basic operations of DE to enhance the global search ability and accelerate the convergence speed. The computational results indicated that the VNSDE is superior to compared algorithms.

Three local search methods with variable neighborhoods, which employs a new neighborhood of exponential size along with the well-known neighborhoods of polynomial size, were developed by Kononova and Kochetov (2013) for obtaining upper bounds of the two machine flow shop problem with passive prefetch. The computational results showed the high efficiency of the developed methods. The parallel VNS and the series VNS were proposed by Ribas and Companys (2013) to solve the block flow shop problem. In the parallel VNS, the insertion and swap neighborhoods are randomly chosen to improve the solution. In the serial VNS, the neighborhoods are executed sequentially. The computational evaluation showed that the serial VNS is competitive. A general VNS (GVNS) was presented by Tasgetiren, Buyukdagli, Pan, and Suganthan (2013) to solve the no-idle permutation flow shop problem. In the GVNS, the VND is employed by the inner loop. The iterated greedy algorithm and iterated local search algorithm are employed in the VND as neighborhood structures. The insert and swap operations are used by the outer loop. The experimental results showed that GVNS outperforms other algorithms on the Ruben Ruiz benchmark. A hybrid variable neighborhood search (HVNS), which combines the VNS and simulated annealing algorithm, was presented by Moslehi and Khorasanian (2014) to solve the limited-buffer permutation flow shop problem (LBPFSP) with the makespan criterion. The computational results demonstrated that the HVNS algorithm is competitive with the algorithms for the blocking flow shop problem. The VNSSA, which incorporates the SA into the framework of VNS, was proposed by Xie, Zhang, Shao, and Yin (2014) to solve the blocking flow shop problem with the total flow time minimization. In the VNSSA, the SA is employed as the local search method in the third stage of VNS, and the best-insert operator is introduced to generate the neighbors in SA. The computational results validate the effectiveness of VNSSA on Taillard's benchmark.

The flow shop problem with two agents was studied by Lei (2015), and the feasibility model was also considered to minimize the makespan of the first agent and the total tardiness of the second agent simultaneously. In this study, a learning neighborhood structure, which combines part of a randomly chosen member and the most parts of current solution, was constructed to produce new solutions in VNS. Besides, the replacement principle is applied to decide if the current solution can be replaced with the new solution. The hybrid harmony search (HHS) was proposed by Zhao et al. (2017) to solve the permutation flow shop problem. In the HHS, a novel VNS which is based on the efficient insert and swap operations was designed to emphasize the local exploitation ability. Experimental simulations demonstrated that the HHS outperforms the compared algorithms in term of solution quality and stability. Four neighborhood structures, which are based on the factory assignment and job sequence adjustment, were incorporated into the framework of VNS by Shao et al. (2017a,b) for solving the distributed NWFSP. The comparisons with the state-of-the-art algorithms demonstrated the effectiveness of proposed algorithms for solving DNWFSP. The iterated local search method and the iterated greedy method were combined with VNS by Ribas, Companys, and Tort-Martorell (2017) for solving the parallel blocking flow shop problem (PBFSP) with the makespan criterion minimization. The computational results demonstrated that the proposed two algorithms are superior to the compared algorithms for solving the PBFSP. The multi-objective parallel variable neighborhood search (MPVNS) was proposed by F. Wang, Deng, Jiang, and Zhang (2018) to solve the blocking flow shop problem. The VNS was designed to explore the initial solutions in parallel. The experimental results illustrated that the MPVNS is superior to the compared multi-objective meta-heuristics.

Although the VNS and its variants have been widely applied to flow shop problems, little attention has been paid to construct VNS with block neighborhood structure for solving the NWFSP. In this paper, a modified VNS based on the block neighborhood structure and insert neighborhood structure is designed to perform exploitation around the current best solutions.

3. No-wait flow shop scheduling problem (NWFSP)

The NWFSP is described as follows: There are n jobs to be processed through m machines, and all jobs have the same processing routes. Every job j ($j = 1, 2, \ldots, n$) has the predefined processing time on every machine i ($i = 1, 2, \ldots, m$). At any moment, a job is only processed at most by one machine and each machine executes no more than one job. Each job is processed without waiting time between consecutive operations. The start of a job is delayed on the first machine to satisfy the no-wait constraint. In this paper, the goal is to find a feasible schedule π which has the minimum makespan for the n jobs.

Assume that $\pi = [\pi(1), \pi(2), \ldots, \pi(n)]$ represents a schedule sequence. Let C_{max} denotes the makespan of π, $p(\pi(i), k)$ is the processing time of the ith job on the kth machine. Then the no wait constraint of problem ensures that the completion time distance between adjacent jobs just related to the processing time of the two jobs. Thus, it can be calculated between each pair of jobs. The completion time distance $D(i, j)$ from job i to job $j$is calculated as follow.

$$D(i,j) = \max_{k=1,\ldots m} \left\{ \sum_{h=k}^{m} (p(j,h) - p(i,h)) + p(i,k) \right\} \tag{1}$$

Then the makespan of the sequence π is obtained as Eq. (2).

$$C_{max}(\pi) = \sum_{i=2}^{n} D(\pi(j-1), \pi(j)) + \sum_{k=1}^{m} P(\pi(1), k) \tag{2}$$

A virtual job, whose processing time is set to zero, is introduced to simplify the calculation process. Sequenceπ is replaced by the $\pi' = [\pi(0), \pi(1), \pi(2), \ldots, \pi(n)]$. Then, the $D(\pi(0), \pi(1)) = \sum_{k=1}^{m} P(\pi(1), k)$. Therefore, the computational formula of makespan is simplified as follows.

$$C_{max}(\pi) = \sum_{i=2}^{n} D(\pi(j-1), \pi(j)) + \sum_{k=1}^{m} P(\pi(1), k)$$

Algorithm 1 The main procedure of original BBO.

```
1. Generate the initial population H.
2. Evaluate the fitness of each individual in H.
3. While the halting criterion is not satisfied do
4.   Calculate λ_i, μ_i and m_i for each individual H_i.
5.     for i = 1 to NP do
6.       for j = 1 to D do
7.         if rndreal(0, 1) < λ_i then // rndreal(0, 1) is a uniform random number on the interval (0,1).
8.           Select H_k with probability ∝ μ_k.
9.           H_i(j) = H_k(j).
10.        end if
11.      end for
12.    end for
13.  for i = 1 to NP do
14.    for j = 1 to D do
15.      if rndreal(0, 1) < m_i then
16.        Replace H_i(j) with a randomly generated SIV.
17.      end if
18.    end for
19.    end for
20. end while
```

$$= \sum_{i=2}^{n} D(\pi(j-1), \pi(j)) + D(\pi(0), \pi(1))$$
$$= \sum_{i=1}^{n} D(\pi(j-1), \pi(j)) \quad (3)$$

Let Π denotes the set of all possible permutation. The minimum makespan is described as follows.

$$C_{max}(\pi^*) = \min\{C_{max}(\pi) | \pi \in \Pi\} \quad (4)$$

Symbols used mostly in this paper are summarized as follows:

N	number of jobs
M	number of machines
π	the sequence of all jobs
π_i	the ith sequence in population
π(j)	the$j$th job in sequence π
$\pi_i(j)$	the$j$th job in ith sequence π_i
$C_{max}(\pi)$	the makespan of π
$p(\pi(i), k)$	the processing time for the $\pi(i)$ on the kth machine
$D(\pi(i-1), \pi(i))$	the delay time between job $\pi(i-1)$ and job $\pi(i)$
NP	the population size
Π	the set of all possible permutation sequence
I	the maximum immigration rate
E	the maximum emigration rate
m_i	the mutation rate of ith sequence
λ_i	the immigration rate of ith sequence
μ_i	the emigration rate of ith sequence
$rmax$	the maximum length of the block
$pmutate$	the maximum mutation rate

4. Biogeography-based optimization

As a novel population-based optimization algorithm, biogeography-based optimization (Simon, 2008) takes inspiration from the science of biogeography. In BBO algorithm, the solution space analogous to habitats in biogeography. The goodness of each habitat is measured by the habitat suitability index (HSI). The habitats with a high HSI tend to accommodate a large number of species, whereas the habitats with a low HSI maintain a few species. New candidates are generated by using migration and mutation. The primary step of the standard BBO is described in Algorithm 1. The leading operators of BBO are briefly described as follows.

4.1. Migration operator

Migration operator is the leading operator in BBO. Features are mixed among habitats based on the immigration rate λ and the emigration rate μ. For the habitats which have a vast number of species, the immigration rate is low because they are already nearly saturated with species. The high HSI habitats have a high emigration rate because the species on high HSI habitats have more opportunities to migrate to other habitats. On the contrary, poor habitats tend to improve their HSI by accepting new features from more attractive habitats in the evolution process. In the standard BBO algorithm, the immigration rate λ_i and emigration rate μ_i are linear functions of the number of species. The linear migration model is calculated using Eqs. (5) and (6).

$$\lambda_i = I \cdot \left(1 - \frac{i}{n}\right) \quad (5)$$

$$\mu_i = E \cdot \left(\frac{i}{n}\right) \quad (6)$$

where I and E are the maximum immigration rate and emigration rate respectively. i is the rank of habitat. n represents the maximum number of species which the habitats accommodated. For convenience, n is equal to population size.

4.2. Mutation operator

As a result of cataclysmic events can drastically change the HSI of a habitat, a mutation operator is employed to modify the features of solutions. Differ from other EAs, the mutation probability is dynamically calculated by Eq. (7).

$$m_i = m_{max} \cdot \left(1 - \frac{p_i}{p_{max}}\right) \quad (7)$$

where m_{max} is the user-defined maximum mutation probability. p_i represents a priori existence probability. $p_{max} = max\{p_i, i = 1, 2, \ldots, ps\}$.

5. The proposed HBV algorithm

In this section, the HBV algorithm for solving the NWFSP with a makespan criterion is presented. In this paper, the job-permutation-based representation, which has been widely used in the literature for the flow shop, is adapted to encode and decode in the mapping process. The HBV consists of the following five

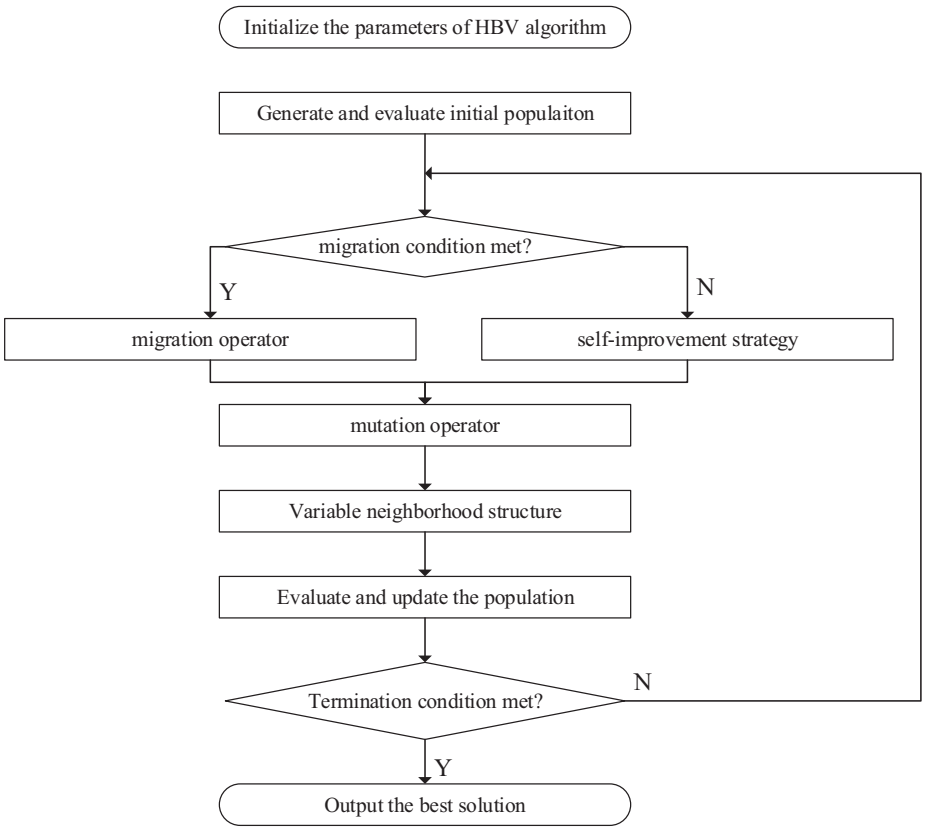

Fig. 1. workflow of HBV for NWFSP.

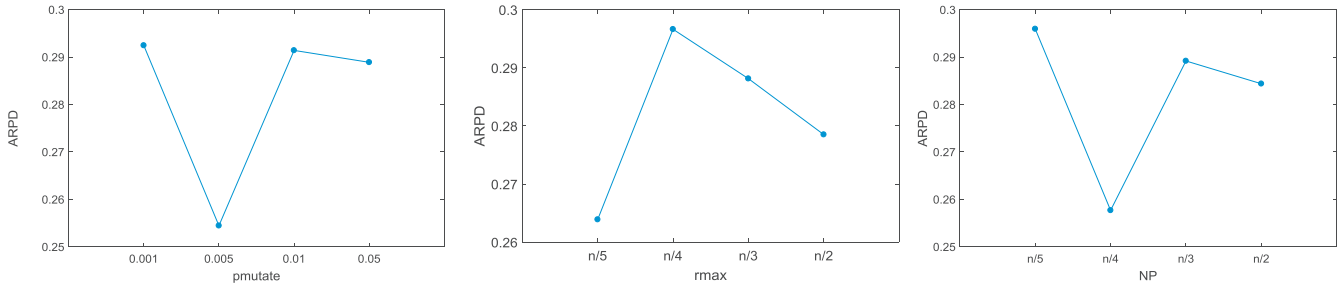

Fig. 2. The trend of parameters.

phases: initial population based on the NN and NEH, the migration operator which is based on the path relink and block-based self-improvement strategy, the mutation operator which is based on the iterated greedy algorithm, the modified variable neighborhood search, and elitism strategy. The details of each phase are fully described in the following sub-sections. The workflow of HBV is also given in Fig. 1.

5.1. Initial population

The nearest neighbor (NN) and the NEH mechanism are two effective constructive heuristics for initial population. The NN and NEH method was combined by Pan et al. (2008a,b) to construct the initial population in the discrete particle swarm optimization algorithm (DPSO). The modified NEH was designed by Gao, Xie, Hua, and Li (2011). The NN+MNEH method was designed by Zhao et al. (2018) to make the initial population. The experimental results show that the NN+MNEH method outperforms NN+NEH and MNEH. In this paper, the NN+MNEH method is employed to construct the initial population.

The detail of NN+MNEH method is described as follows. Firstly, pick out NP jobs randomly from a set $S_0 = \{J_1, J_2, \ldots, J_n\}$, which contains all the jobs, to construct a sequence $JF_s = \{JF_1, JF_2, \ldots JF_n\}$. Then the ith jobs of JF_s is taken as the first job of ith candidate sequence. Secondly, NN is employed to find the second job. Thirdly, the MNEH is applied to construct a partial sequence with other $(n-2)$ jobs. The partial sequence is appended to the first scheduled jobs to construct the final permutation. The pseudo code of initial population is given in Algorithm 2.

5.2. Migration operator

The migration strategy, which is the most significant operator in BBO, is similar to the global combination approach of other evolutionary algorithms. The migration operator is employed to modify existing solutions instead of creating new solutions. The

Algorithm 2 Initial population.

1. Pick up NP jobs from a set S_0 and construct a sequence JF_s
2. **for** $k = 1$: NP
3. $\pi_k(1) = JF_k$
4. $\pi_k(2) = NN(\pi_k(1))$
5. $S_1 = S_0 - \{\pi_k(1), \pi_k(2)\}$
6. $\pi_k = [\pi_k(1), \pi_k(2)] + MNEH(S_1)$
7. $H(k) = \pi_k$
8. **end for**

Algorithm 3 Path relink technique.

1. for $k = 1$: NP **do**
2. **if** $H_i \neq H_e$ **do**
3. find the position s of $H_e(k)$ in H_i
4. swap $(H_i(k), H_i(s))$ and generate intermediate solution H'
5. **end if**
6. find the solution H_{best} with lowest makespan
7. $H_i = H_{best}$
8. end for

migration characteristics of sinusoidal migration model are closer to natural law as Ma (2010) analyzed. Therefore, the sinusoidal migration model outperforms the linear migration model in the standard BBO algorithms. In this paper, the sinusoidal migration model is employed to calculate the immigrate rate and the emigrate rate of each in population. The sinusoidal model is given in Eqs. (8) and (9).

$$\lambda_k = \frac{I}{2}\left(\cos\frac{k\pi}{n} + 1\right) \tag{8}$$

$$\mu_k = \frac{E}{2}\left(-\cos\frac{k\pi}{n} + 1\right) \tag{9}$$

The immigration rate and the emigration rate of each solution are employed to share information between habitats probabilistically. In this paper, if a given solution is selected to be modified, the corresponding immigration rate λ is used to decide whether or not to perform the migration operator probabilistically. If the solution is selected to immigrate, another solution is employed to share features with the immigration solution by the emigration rate. Otherwise, the self-improvement strategy, which is based on the block neighborhood structure, is designed to improve the quality of the solution.

Path relink technique is employed to perform the migration operator in HBV. The detail of the migration operator, which is based on the path relink technique, is described as follows. For the emigration habitat H_e and the immigrate habitat H_i, a series of swap operator are performed to transform H_i to H_e. Thus, an intermediate solution is obtained through each swap operator. If the intermediate solution is different from both the immigrate habitat H_i and the emigrate habitat H_e, the intermediate solution is put into a set S. After all the solution in S are evaluated, immigrate habitat H_i is replaced by the intermediate solution with the lowest makespan. The pseudo code of path relink technique is given in Algorithm 3.

The block neighborhood structure is a simple and effective neighborhood structure which is proposed by Ding et al. (2015a,b). In this paper, the block neighborhood structure is integrated into the self-improvement strategy. The self-improvement strategy is performed while a solution is not selected to immigrate. The main procedure of self-improvement strategy is described in Algorithm 4.

Algorithm 4 Self-improvement strategy.

1. Set$bestFit$ = inf and $bestSeq = \emptyset$
2. for $k = 1$: $(n - r)$ **do**
3. remove a block from $H_i(k)$ to $H_i(k + r)$
4. find the sequence H' by inserting the block in any possible position
5. **if** $f(H') < bestFit$ **do**
6. $bestFit = f(H')$
7. $bestSeq = H'$
8. **end if**
9. end for
10. $H_i = bestSeq$

5.3. Mutation operator

Mutation operator, which is employed to expand the search space and escape from the local optimum solutions, is the main operator in the BBO. In standard BBO, the mutation operator is performed by simply replacing the solution with a new solution which is generating randomly. In this paper, the iterated greedy (IG) algorithm is integrated into the mutation operator to obtain a promising solution in exploitation phase.

IG is a simple and effective algorithm for solving the combinatorial optimization problem. The new solution is obtained by iterated applying destruction operation and construction operation in IG. In the destruction phase, d jobs are selected randomly and removed from the candidate solution π. Therefore, π is divided into two parts π_d and π_r. π_d contains d jobs and π_r contains $(n\text{-}d)$ jobs. In the construction phase, each job in π_d is inserted into π_r sequentially. The pseudo code of IG is given in Algorithm 5.

5.4. Variable neighborhood search

Variable neighborhood search (VNS) is a popular meta-heuristic proposed by Hansen and Mladenović (1997) systematically exploiting the idea of neighborhood change. The principle can be described as follows.

- A local minimum concerning one neighborhood structure is not necessarily so for another.
- A global minimum is a local minimum concerning all possible neighborhood structures.
- For many problems, local minima concerning one or several neighborhoods are relatively close to each other.

The VND, which is a variant of VNS, is embedded as a local search algorithm in the DPSO algorithm to obtain better results by Pan et al. (2008a,b). The *insert* + *swap* neighborhood structure is integrated into VND in DPSO. As Ding et al. (2015a,b) pointed out, the block-based operator explores a lager neighborhood solution space than traditional insert and swap operators. In this paper, a modified VNS which is presented by slightly modifying is only applied to the global best solution G^t. The swap neighborhood is replaced by the block neighborhood. The modified VNS is described as Algorithm 6.

5.5. Elitism strategy

BBO with only migration and mutation does not converge to a global optimal as Ma, Dan, and Fei (2014) analyzed. In this paper, the elitism strategy is adapted to retain the best solution in the population. If the fitness value of the current optimal solution is larger than the fitness value of the global optimal solution of the previous generation at the end of each generation, the inferior solution in population is replaced by the global optimal solution of the previous generation.

Algorithm 5 Iterated greedy algorithm.

1. $\pi_r = \pi$
2. for $k = 1$: d **do**
3. remove one job from π_r and put it into π_d
4. end for
5. for $k = 1$: d **do**
6. insert job $\pi_d(k)$ into the optimal location with the minimum makespan in π_r
7. end for

Algorithm 6 Variable neighborhood search.

1. $s_0 = G^t$
2. s = perturbation (s_0)
3. s_1 = insert (s)
4. s_2 = block neighborhood search (s1)
5. **if** $f(s_2) < f(s_0)$ **do**
6. $G^t = s_2$
7. **end if**

Algorithm 7 The main procedure of HBV.

1. Generate the initial population H.
2. Evaluate the HSI for each habitat in H.
3. **While** the halting criterion is not satisfied **do**
4. For each individual map the fitness to the number of species.
5. **for** i = 1: NP **do**
6. Calculate the immigration rate λ_i and the emigration rate μ_i.
7. **if** *rand* $< \lambda_i$ **do**
8. Select H_e with the emigration rate μ_i.
9. U_i=Path relink (H_i,H_e).
10. **else**
11. U_i=Block neighborhood search (H_i).
12. **end if**
13. **if** rand$< m_i$ **do**
14. Modify the population U with IG shown in Algorithm 5.
15. **end if**
16. **end for**
17. Evaluate the fitness of each individual in U.
18. $H = U$.
19. Sort H with the *HSI*.
20. **if** $f(H_1) >$ *bestFit* **do**
21. H_1 = bestSeq.
22. **end if**
23. $H_1 = VNS(bestSeq)$.
24. **end while**

5.6. The framework of HBV

This paper presents a hybrid biogeography-based optimization with variable neighborhood search (HBV) for the NWFSP. The HBV has five phases: (i) initial population, (ii) migration operator, (iii) mutation operator, (iv) variable neighborhood search, (v) elitism strategy. The pseudo-code of HBV is summarized in Algorithm 7. Suppose that there are n jobs and m machines in the NWFSP, the population size is NP, the mutation probability is p. Since the population initialization is based on the NEH mechanism, the computational complexity of population initialization is given as $O(n^2)$. The migration operator consists of two parts: path relink and self-improvement strategy. The migration operator executes Np times fitness evaluations in each generation. The mutation operator probably employs the IG mechanism to search around the potential solutions. The VNS employs the insert and block neighborhood structure to search around the best solution. As the speed-up method, the complexity of fitness evaluation is $O(1)$ when a sequence is generated by insert, swap or block operator. In each generation, the computational complexity of migration operator is $O(mn)$. The computational complexity of mutation operator is $O(n^2)$. The computational complexity of variable neighborhood search is $O(2n^2)$. The computational complexity of elitism strategy is $O(1)$. Suppose the number of generations is k, the computational complexity of HBV algorithm is calculated as follows.

$$\begin{aligned} O(NP, p, n, m) &= O\left(n^2\right) + O(k) \\ &\quad \times \left[O(mn) + O\left(n^2\right) + O\left(2n^2\right) + O(1)\right] \\ &\approx O\left(n^2\right) + O(k) \times O\left(2n^2\right) \\ &\approx O\left(2kn^2\right) \end{aligned}$$

5.7. Convergence analysis of HBV

In this section, the convergence behavior of HBV is analyzed with Markov model.

According to the definition of NWFSP, the search space of the problem is represented as $\boldsymbol{I} = \boldsymbol{n}!$. The HBV algorithm with population size NP consists of migration, mutation, elitism and VNS. Some definitions are given as follows.

Theorem 1. (Convergence in probability) (Burton, 1985) *$\{\boldsymbol{A}(\boldsymbol{t}), \boldsymbol{t} = 0, 1, 2\ldots\}$ is a population sequence generated by a population-based stochastic algorithm, the stochastic sequence $\{\boldsymbol{A}(\boldsymbol{t})\}$ weakly converges in Probability to the global optimum. If and only if:*

$$\lim_{t\to\infty} P\{A(t) \cap I^* \neq \emptyset\} = 1 \tag{10}$$

where I^* is the set of global optimal of a problem.

Theorem 2. *$\boldsymbol{P}$ is a stochastic matrix with the structure $\boldsymbol{P} = \begin{bmatrix} \boldsymbol{C} & 0 \\ \boldsymbol{R} & \boldsymbol{Q} \end{bmatrix}$, where $\boldsymbol{C}$ is a primitive stochastic matrix and $\boldsymbol{R}$, $\boldsymbol{Q} \neq 0$. $\boldsymbol{P^k}$ converges to a stochastic matrix as $\boldsymbol{k} \to \infty$,. That is,*

$$P^{\infty} = \lim_{k\to\infty} \left(p_{ij}\right)^{(k)} = \begin{pmatrix} \pi \\ \vdots \\ \pi \end{pmatrix}_{T\times T} \tag{11}$$

where $\boldsymbol{\pi} = (\boldsymbol{\pi}_1, \ldots \boldsymbol{\pi}_{\boldsymbol{m}}, 0, \ldots, 0)$, and $\boldsymbol{\pi_j} \neq 0$ for $1 \leq \boldsymbol{j} \leq \boldsymbol{m} < \boldsymbol{T}$

Definition 1. $\boldsymbol{A}(\boldsymbol{t}) = \{\boldsymbol{a_i}(\boldsymbol{t}) | \boldsymbol{i} \in [1, \boldsymbol{N}], \ \boldsymbol{a_i}(\boldsymbol{t}) \in \boldsymbol{I}\}$ is the population at the generation $\boldsymbol{t}$. $\boldsymbol{a_i}(\boldsymbol{t})$ is a candidate solution in search space $\boldsymbol{I}$. Then let $\boldsymbol{I}^*$ denote a set of global optimal of an optimization problem. The best individuals in the population at generation $\boldsymbol{t}$ are $\boldsymbol{a}^*(\boldsymbol{t})$. Due to $\boldsymbol{A}(\boldsymbol{t})$ may contain duplicate elements, let $\boldsymbol{I}^*(\boldsymbol{t})$ represent the set of $\boldsymbol{a}^*(\boldsymbol{t})$.

Obviously, $a^*(t)$ changes randomly over time. As $t \to \infty$, the convergence of $a^*(t)$ indicates the globally convergent of HBV algorithm. That is, HBV is said to converge if

$$\mathrm{P}\left(\lim_{t\to\infty} a^*(t) \in I^*\right) = 1 \Leftrightarrow \mathrm{P}\left(a^* \in \lim_{t\to\infty} A(t)\right) = 1 \tag{12}$$

Thus, the evolution of $a^*(t)$ is a homogeneous finite Markov chain, namely, $a^*(t)$-chain.

Definition 2. Let $\hat{\boldsymbol{P}}^{\boldsymbol{t}} = (\hat{p}_{ij})$ be the transition matrix of an $a^*(t)$-chain, where $\hat{p}_{ij}$ is the probability that $a^*(t) = I_i$ transition to $a^*(t+1) = I_j$. The HBV algorithm converges to global optimum if $a^*(t)$ transition from any state $i \in I$toI^* as $t \to \infty$ with probability 1, that is, if

$$\lim_{t\to\infty} \sum_{I_j \in I^*} \left(\hat{P}^t\right)_{ij} = 1 \quad \forall\ i,\ I_i \in I \tag{13}$$

Theorem 3. *If the transition probability* $\hat{\boldsymbol{P}} = (\hat{\boldsymbol{p}}_{\boldsymbol{ij}})$ *of an* $\boldsymbol{a}^*(\boldsymbol{t})$*-chain is a stochastic matrix with the structure* $\hat{\boldsymbol{P}} = \begin{pmatrix} \boldsymbol{C} & 0 \\ \boldsymbol{R} & \boldsymbol{Q} \end{pmatrix}$*, where C is a positive stochastic matrix, and* $\boldsymbol{R}, \boldsymbol{Q} \neq 0$*, then the HBV algorithm converges to one or more of global optima.*

Proof. From Theorem 3, for any $i \in S$,

$$\lim_{t\to\infty} \sum_{I_j \in I^*} \left(\hat{P}^t\right)_{ij} = \sum_{I_j \in I^*} \lim_{t\to\infty} \left(\hat{P}^t\right)_{ij} = \sum_{j=1}^{|I^*|} \pi_j = 1 \tag{14}$$

Theorem 4. *In HBV algorithm, the transition probability of* $\boldsymbol{a}^*(\boldsymbol{t})$*-chain satisfied the following conditions:* $\forall \boldsymbol{k} > \boldsymbol{i}$, $\boldsymbol{p_{ij}} = 0$ and $\forall \boldsymbol{k} < \boldsymbol{i}$, $\boldsymbol{p_{ij}} \geq 0$.

Proof. As above definition, $A(t)$ denotes the population at the generation t, $a^*(t)$ is the best individual in the population generation t. In each generation of HBV, the migration operator, mutation operator, elitism strategy and VNS are performed conditionally. Then let F_{best} denotes the current best solution after the migration mutation operator. If $F_{best} < f(a^*(t-1))$, The elitism strategy is not implemented and $a^*(t) = F_{best}$, otherwise the elitism strategy is performed,$a^*(t) = a^*(t-1)$. In the VNS operator, the $a^*(t)$ will be replaced by one with better fitness, otherwise it will remain unchanged. Obviously, the $a^*(t)$-chain is monotonous and $\forall k > i$, $p_{ij} = 0$. Apart from this, the new solution can be obtained by migration operator and mutation operator. After migrate and mutate $t\,(t \to \infty)$ generations, $(P_r)^t > 0$. That is, the probability of searching for any state k in the space from the initial state i after t generation is $p_{ik}^{\infty} > (P_r)^t > 0$. Then, $\forall k < i$, $p_{ij} \geq 0$.

If the best individual in HBV is equal to global optimum, it is an absorbing state of the $a^*(t)$-chain, According to Theorem 4, the best individual can only be replaced by one with better fitness. Then, the $a^*(t)$-chain of HBV contains three classes of states:

(1) At least one absorbing state.
(2) Non-absorbing states which transition to absorbing states in one step.
(3) Non-absorbing states which transition to non-absorbing states in one step.

Thus, the transition matrix P is given as $\hat{P} = \begin{pmatrix} C & 0 \\ R & Q \end{pmatrix}$. C is the matrix corresponding to optimal individuals. R is a matrix corresponding to the non-absorbing states which transition to absorbing states in one step. Q is the matrix corresponding to the non-absorbing states which transition to non-absorbing states. The HBV algorithm converges to one or more global optima as the Theorem 3.

6. Numerical result

In this section, the HBV algorithm is compared with the state-of-the-art algorithms to test the performance for solving the NWFSP with minimization of makespan. 600 instances are employed to test the considered algorithms: (i) The 120 Taillard's instances provided by Taillard (1993). Ta001 -Ta120 consists of 12 subsets of different size which is ranging from 20 jobs and 5 machines to 500 jobs and 20 machines. (ii) 480 instances provided by Vallada, Ruiz, and Framinan (2015). This benchmark problem set consists of 240 large instances and 240 small instances. The small instances include 24 combinations of $n = \{10, 20, 30, 40, 50, 60\}$ jobs with $m = \{5, 10, 15, 20\}$ machines. The large instances include 24 combinations of $n = \{100, 200, 300, 400, 500, 600, 700, 800\}$ jobs with $m = \{20, 40, 60\}$ machines. For each size of problems, ten instances are provided. The optimal solutions are provided by Lin and Ying (2016). In order to make a fair comparison, all the algorithms were coded using MATLAB. The simulation experiments were carried out on a personal computer (PC) with Intel (R) Core (TM) i7-6700 CPU 3.4 GHz and 8.00GB memory with a Windows Server 2012 Operating System.

Table 1
The parameters for different levels.

Parameters	Levels			
	1	2	3	4
pmutate	0.001	0.005	0.01	0.05
rmax	*n*/5	*n*/4	*n*/3	*n*/2
NP	*n*/5	*n*/4	*n*/3	*n*/2

The average relative percentage deviation (ARPD) is applied to measure the quality of the experimental results and compare the performance of HBV with other algorithms visually. The value of ARPD is calculated as follows:

$$ARPD = \frac{1}{R}\sum_{r=1}^{R} \frac{C_r - C_R{}^*}{C_R{}^*} \times 100 \tag{15}$$

where C_R^* is the best solution found so far, C_r is the solution of the rth experiment and R is the number of runs. The less value of ARPD is, the better the performance of algorithm is. The standard deviation (SD) is also applied to indicate the robust of the algorithms. The value of SD is calculated as follows:

$$SD = \sqrt{\frac{\sum_{r=1}^{R} \left(C_r - \bar{C}\right)^2}{R}} \tag{16}$$

where $\bar{C}$ is the mean value of Rsolutions.

This experimental section consists of five parts. Section 6.1 provides the analysis of parameters. The effectiveness of the components is analyzed in Section 6.2. In Section 6.3, the proposed HBV algorithm is compared with existing state-of-the-art algorithms on Taillard's instances. The comparisons of the considered algorithms on the VRF_hard instances are given in Section 6.4. The experimental results are analyzed in Section 6.5.

6.1. Analysis of parameters

The appropriate design of parameters plays an essential role in the performance of HBV. In HBV, the maximum migration rate is set to 1. There are three crucial parameters: the maximum probability of mutation *pmutate*, the maximum length of block *rmax* and the population size *NP*. The Taguchi method (Montgomery, 1976) of design for the experiments is introduced to investigate the best parameter setting for HBV. 240 small-scale instances and 48 large scale instances in VRF benchmark which is provided by Vallada et al. (2015) were employed to calibrate parameters. The various values of parameters are listed in Table 1. n is the job size. The different parameter combinations and ARPD are given in Table 2.

From Table 3, the NP is the most significant parameter of HBV as the HBV is a population-based algorithm. The parameter *pmutate* ranks the second place, which illustrates that *pmutate* is also an important factor in HBV. A large *pmutate* diversifies the population significantly, and a small *pmutate* accelerate the convergence speed of HBV. The *rmax* ranks the third place, which demonstrates that the parameter *rmax* has slight effect on the performance of HBV. According to the above analysis, the parameters in HBV are suggested as follows. $pmutate = 0.005$, $rmax = n/5$, $NP = n/4$.

6.2. Effectiveness of algorithm components

The HBV without VNS and HBV without self-improvement were compared with the HBV algorithm to test the effectiveness of the

Table 2
Parameter combinations and average makespan.

No.	Parameter combination			ARPD
	pmutate	*rmax*	*NP*	
1	1	1	1	0.304608
2	1	2	2	0.298258
3	1	3	3	0.289861
4	1	4	4	0.277049
5	2	1	2	0.166872
6	2	2	1	0.297946
7	2	3	4	0.279021
8	2	4	3	0.27391
9	3	1	3	0.299111
10	3	2	4	0.296481
11	3	3	1	0.294051
12	3	4	2	0.275929
13	4	1	4	0.284951
14	4	2	3	0.293834
15	4	3	2	0.289608
16	4	4	1	0.287131

Table 3
Parameter rank and response value.

Levels	*pmutate*	*rmax*	*NP*
1	0.2924	0.2639	0.2959
2	0.2544	0.2966	0.2577
3	0.2914	0.2881	0.2892
4	0.2889	0.2785	0.2844
Response value	0.038	0.0327	0.0383
Rank	2	3	1

Table 4
Computational result of HBV without VNS, HBV without self-improvement and HBV.

$n \times m$	HBV_NV		HBV_NS		HBV	
	ARPD	SD	ARPD	SD	ARPD	SD
20 × 5	0.02	0.19	0.00	0.12	**0.00**	**0.00**
20 × 10	0.08	0.35	0.00	0.09	**0.00**	**0.06**
20 × 20	0.02	0.74	**0.00**	**0.00**	**0.00**	**0.00**
50 × 5	0.19	1.92	0.49	1.39	**0.15**	**1.02**
50 × 10	0.12	1.11	0.34	2.15	**0.10**	**0.96**
50 × 20	0.11	2.39	0.27	2.44	**0.06**	**0.45**
100 × 5	0.30	3.37	1.56	3.84	**0.28**	**1.67**
100 × 10	0.25	3.07	0.99	4.00	**0.24**	**2.27**
100 × 20	0.25	3.03	0.84	6.61	**0.21**	**2.58**
200 × 10	0.58	4.07	2.46	7.96	**0.57**	**3.88**
200 × 20	0.56	7.04	1.89	13.94	**0.47**	**5.45**
500 × 20	1.20	10.34	3.67	26.58	**1.04**	**10.13**
Average	0.31	3.13	1.04	5.76	**0.26**	**2.37**

algorithm components. Table 4 lists the computational results obtained in the three variants. ARPD and SD denote the average relative percentage deviation and standard deviation for 30 times running respectively.

From Table 4, the HBV outperforms HBV algorithm without VNS (HBV_NV) and the HBV without self-improvement (HBV_NS), which illustrates that the VNS and the self-improvement significantly improve the performance of HBV. In addition, the performance of HBV without VNS is superior to HBV without self-improvement strategy. The reason may be that self-improvement enhances the exploration ability of HBV to avoid falling into local optimal. From Table 4, the HBV obtains better results than other variants. As above analysis, the HBV balance exploration and exploitation ability effectively.

6.3. Results and comparisons for Taillard's benchmark

For the Taillard's instances, the proposed HBV algorithm was compared with five existing state-of-the-art algorithms including IIGA (Pan et al., 2008a,b), TMIIG (Ding et al., 2015a,b) DWWO (Zhao et al., 2018), mPTLM (Shao et al., 2017a,b) and CGLS (Riahi, Newton, Su, & Sattar, 2019). The compared algorithms are chosen as the HBV is a population-based algorithm which is directly applied to solve the NWFSP. Among the compared algorithms, the mPTLM is a population-based algorithm which employs the large-order-value (LOV) rule to build the mapping from the continuous variables to the job sequence. Compared with the mPTLM, the advantage of job-permutation-based representation which is employed by HBV is shown. Besides, the DWWO is also a population-based algorithm which employs the job-permutation-based representation. The performance of HBV algorithms is demonstrated by comparing with DWWO. The TMIIG and IIGA are also compared with HBV to indicate the difference between the population-based algorithms and the adaptable algorithms. In addition, the CGLS, which is an objectively adapted algorithm for the BFSP, is also carried to compare with HBV. The compared algorithms are re-implemented as all the details given in the original papers to make a fair comparison. Meanwhile, all the algorithms were dependently run with the maximum running time $(n/2) \times m \times \rho\ ms$, where $\rho = \{5, 15, 30\}$. Each instance was run for 30 times. The computational results are summarized in Tables 5–7 respectively where the best results are given in bold.

Several conclusions are contained by the results of experiment which shown in Tables 5–7. The proposed HBV performs less average value of ARPD on most of the Taillard's instances $\rho = \{5, 15, 30\}$. For the short running times $\rho = 5$ and $\rho = 15$, the value of ARPD obtained by HBV are 0.58 and 0.34 respectively, which is significantly smaller than the other compared algorithm. The smaller ARPD indicates that the HBV obtain better solutions within a small range of time. The larger value of SD, which is larger than the mPTLM, demonstrated that the population diversity is controlled effectively. For the running time $\rho = 30$, the ARPD value obtained by HBV is 0.27, which is less than 0.95, 0.6, 0.92, 1.77 obtained by other compared algorithms respectively. The Fig. 3 shows the interval plot for the interaction between the algorithms and the maximum running time. From Fig. 3, the result obtained by HBV significantly better than the other compared algorithms.

Two rigorous statistical studies based on the ARPD value for 120 Taillard's instances are employed to investigate whether the results of algorithms are rather significant for solving NWFSP with the objective of minimization makespan criterion. The multiple-problem Wilcoxon's test is performed to demonstrate the performance of the above six algorithms. As the statistical analysis results listed in Table 8, the HBV provides higher $R+$ values than $R-$ values in all cases. The result demonstrates that the HBV is significantly better among the compared algorithms for solving NWFSP problems with $\alpha = 0.05$, $\alpha = 0.01$.

The Friedman's test is carried out to further detect the significant differences between HBV and other compared algorithms. Table 9 summarizes the average ranking of the five algorithms obtained by the Friedman's test. HBV has the best ranking among the six algorithms. Therefore, the Bonferroni–Dunn's method is applied as a post hoc procedure to calculate the critical difference for comparing the differences of compared algorithms with $\alpha = 0.05$, $\alpha = 0.01$. From Fig. 4, the solutions obtained by HBV algorithm are significantly better than the solutions obtained by other compared algorithms with $\alpha = 0.05$ and $\alpha = 0.01$ on Taillard's benchmark.

For the running time $\rho = 30$, the values of SD obtained by the six algorithms are also calculated to demonstrate the robustness of the HBV algorithm. The average value of SD obtained by HBV is 10.4, which is less than the 15.51, 17.99, 18.58 and 33.88 obtained by DWWO, TMIIG, IIGA and CGLS respectively. The value of SD obtained by HBV is slightly larger than the value of SD obtained by mPTLM. The boxplots of Ta057, Ta067, Ta097 and Ta117 are given in Fig. 6. The robust of HBV is better than DWWO, TMIIG, IIGA, CGLS

Table 5
Computational results of Taillard's benchmark with $\rho = 5$.

Instance	$n \times m$	HBV		DWWO		mPTLM		TMIIG		IIGA		CGLS	
		ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD
Ta01-10	20 × 5	**0.00**	**0.09**	0.06	1.21	0.01	0.25	0.02	0.51	0.02	0.34	0.17	1.21
Ta11-20	20 × 10	0.01	0.14	0.09	2.26	**0.00**	**0.09**	0.02	0.60	0.01	0.39	0.26	1.31
Ta21-30	20 × 20	**0.00**	**0.00**	0.03	0.22	0.01	0.39	0.03	0.38	0.02	0.32	0.19	1.37
Ta31-40	50 × 5	**0.30**	4.86	0.81	6.13	0.42	**4.76**	0.74	7.77	1.72	6.71	1.48	11.04
Ta41-50	50 × 10	**0.19**	**4.81**	0.55	8.08	0.22	5.94	0.47	8.87	0.81	7.27	0.86	10.89
Ta51-60	50 × 20	**0.11**	**5.36**	0.49	13.09	0.36	12.78	0.43	9.81	0.60	9.89	0.82	17.06
Ta61-70	100 × 5	**0.65**	8.63	3.33	13.81	1.66	**7.96**	2.02	14.60	3.51	12.78	2.52	23.22
Ta71-80	100 × 10	**0.45**	11.81	1.20	17.43	0.64	**8.89**	1.45	18.98	2.10	16.23	2.02	36.08
Ta81-90	100 × 20	**0.35**	12.70	0.83	16.13	0.50	**12.62**	1.18	23.59	1.80	17.89	2.15	36.96
Ta91-100	200 × 10	**1.37**	26.67	3.26	31.04	1.77	**18.26**	3.17	37.45	3.62	25.75	3.61	62.10
Ta101-110	200 × 20	**0.88**	29.32	1.83	31.25	0.97	**22.20**	2.25	35.78	2.95	36.78	2.76	71.86
Ta111-120	500 × 20	2.62	64.98	3.64	64.13	**2.08**	**55.02**	4.33	66.72	4.61	85.25	4.67	77.19
Avg		**0.58**	14.12	1.34	17.06	0.72	**12.43**	1.34	18.75	1.82	18.30	1.79	29.19

Table 6
Computational results of Taillard's benchmark with $\rho = 15$.

Instance	$n \times m$	HBV		DWWO		mPTLM		TMIIG		IIGA		CGLS	
		ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD
Ta01-10	20 × 5	**0.00**	**0.03**	0.02	0.67	0.00	0.12	0.00	0.17	0.00	0.14	0.14	1.00
Ta11-20	20 × 10	0.01	0.13	0.03	0.62	0.01	0.44	**0.00**	**0.09**	0.00	0.21	0.19	1.10
Ta21-30	20 × 20	**0.00**	**0.00**	0.02	0.38	0.02	0.39	0.02	0.44	0.02	0.37	0.15	1.50
Ta31-40	50 × 5	**0.21**	3.80	0.60	5.92	0.35	**3.67**	0.52	5.42	1.60	7.82	1.26	11.26
Ta41-50	50 × 10	**0.13**	4.01	0.44	7.93	0.16	**3.77**	0.31	6.39	0.79	6.78	0.71	10.09
Ta51-60	50 × 20	**0.08**	**3.95**	0.31	10.54	0.27	11.54	0.29	9.44	0.60	9.66	0.74	16.00
Ta61-70	100 × 5	**0.37**	**6.97**	2.94	15.69	1.53	8.01	1.53	13.70	3.39	13.12	2.13	18.86
Ta71-80	100 × 10	**0.28**	8.72	0.86	12.80	0.55	**7.29**	1.08	17.70	2.09	15.18	1.67	27.07
Ta81-90	100 × 20	**0.26**	**10.89**	0.64	15.42	0.42	12.54	0.93	19.97	1.74	21.56	1.84	41.71
Ta91-100	200 × 10	**0.74**	19.65	2.64	28.08	1.67	**17.13**	2.60	32.12	3.59	24.30	3.07	66.90
Ta101-110	200 × 20	**0.58**	23.00	1.40	28.36	0.90	**18.15**	1.80	31.30	2.96	38.84	2.36	65.51
Ta111-120	500 × 20	**1.42**	49.47	2.91	66.95	1.94	**44.51**	3.75	73.22	4.57	90.51	4.38	125.47
Avg		**0.34**	10.88	1.07	16.11	0.65	**10.63**	1.07	17.50	1.78	19.04	1.55	32.21

Table 7
Computational results of Taillard's benchmark with $\rho = 30$.

Instance	$n \times m$	HBV		DWWO		mPTLM		TMIIG		IIGA		CGLS	
		ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD
Ta01-10	20 × 5	**0.00**	**0.00**	0.01	0.09	0.00	0.03	**0.00**	**0.00**	0.00	0.15	0.11	1.18
Ta11-20	20 × 10	**0.00**	**0.00**	0.01	0.15	0.00	0.06	0.00	0.06	0.00	0.15	0.14	1.32
Ta21-30	20 × 20	**0.00**	**0.00**	0.01	0.45	0.00	0.24	0.01	0.39	0.02	0.32	0.13	2.23
Ta31-40	50 × 5	**0.16**	**3.66**	0.50	5.26	0.29	4.18	0.37	4.64	1.53	6.99	1.01	11.06
Ta41-50	50 × 10	**0.07**	**2.62**	0.39	5.56	0.13	3.67	0.22	4.70	0.75	8.69	0.58	10.13
Ta51-60	50 × 20	**0.06**	**3.13**	0.31	10.88	0.19	8.61	0.22	7.95	0.59	9.30	0.65	17.38
Ta61-70	100 × 5	**0.29**	**6.68**	2.65	12.80	1.42	8.15	1.26	12.18	3.44	10.96	1.83	16.68
Ta71-80	100 × 10	**0.23**	7.76	0.74	11.62	0.48	**5.79**	0.86	15.11	2.04	14.58	1.48	25.93
Ta81-90	100 × 20	**0.21**	10.21	0.56	13.91	0.37	**8.80**	0.81	19.51	1.78	19.67	1.63	40.54
Ta91-100	200 × 10	**0.55**	17.72	2.34	30.55	1.62	**15.93**	2.18	28.42	3.56	31.55	2.77	70.86
Ta101-110	200 × 20	**0.49**	22.65	1.22	24.57	0.80	**14.68**	1.60	30.86	2.98	32.80	2.11	67.39
Ta111-120	500 × 20	**1.12**	50.35	2.69	70.20	1.87	**38.56**	3.46	91.98	4.51	87.82	4.14	141.87
Avg		**0.27**	10.40	0.95	15.50	0.60	**9.06**	0.92	17.99	1.77	18.58	1.38	33.88

Table 8
Results of the multiple-problem Wilcoxon's test on Ta instances at $\alpha = 0.05$ and $\alpha = 0.01$ significance level.

HBV vs.	$R+$	$R-$	Z	p-value	$\alpha = 0.05$	$\alpha = 0.01$
DWWO	4656.00	0.00	−8.51	1.78E−17	Yes	Yes
mPTLM	4252.00	26.00	−8.37	1.90E−16	Yes	Yes
TMIIG	4276.00	2.00	−8.32	8.69E−17	Yes	Yes
IIGA	4560.00	0.00	−8.46	2.60E−17	Yes	Yes
CGLS	5883.00	3.00	−9.01	2.03E−19	Yes	Yes

and similar to mPTLM for Taillard's benchmark. The convergence curves of the above algorithms for solving Ta057, Ta067, Ta097 and Ta117 are also given in Fig. 5. Compared with the convergence speed of DWWO, TMIIG, IIGA and mPTLM, the convergence speed of HBV is fastest on Taillard's benchmark.

6.4. Results and comparisons for VRF's benchmark

In the above experiment, the Taillard's benchmark has been employed to evaluate the performance of compared algorithms for solving the NWFSP. Since the Taillard's instances were tackled by various researchers, the VRF benchmark which was proposed by Vallada et al. (2015) for PFSP is carried out to test the performance of the proposed HBV in this paper. The VRF's benchmark consists

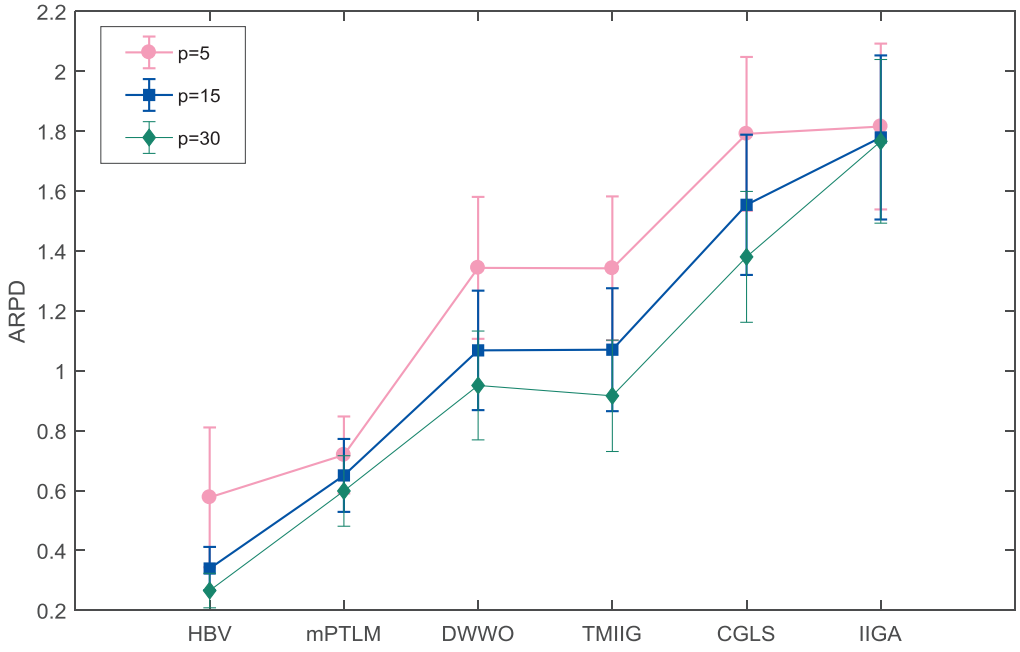

Fig. 3. The interval plot for the interaction between the algorithms and the maximum running time ρ (Taillard's benchmark).

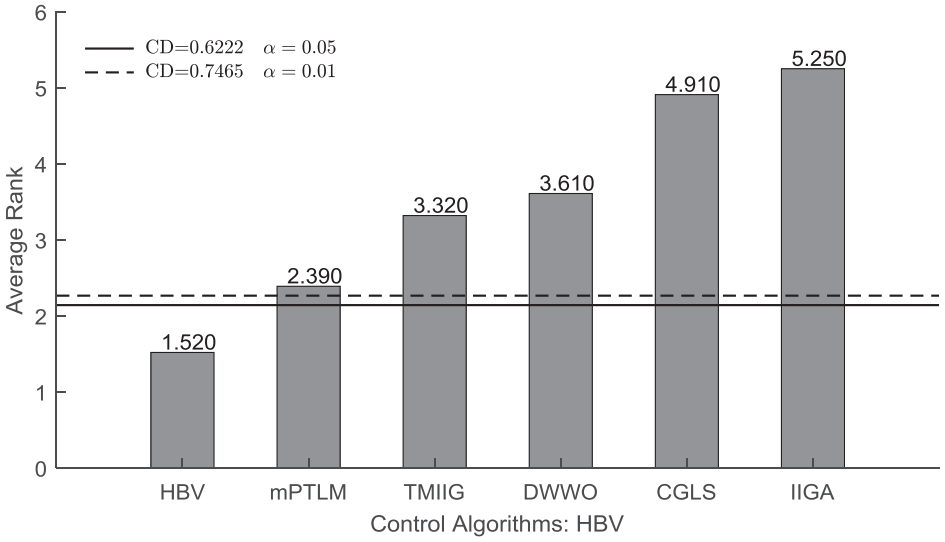

Fig. 4. Rankings obtained through the Friedman test and graphical representation.

Table 9
Ranking of algorithms obtained through Friedman's test.

Algorithm	Mean rank	Chi-Square	*p*-value
HBV	1.52	416.59	7.88E−88
DWWO	3.61		
mPTLM	2.39		
TMIIG	3.32		
IIGA	5.25		
CGLS	4.91		
Crit. Diff. $\alpha = 0.05$	0.6222		
Crit. Diff. $\alpha = 0.01$	0.7465		

of a small benchmark set and a large benchmark set. The small benchmark set, which is denoted by VRF_hard_small benchmark, consists of 240 small instances with the combinations of $n = \{10, 20, 30, 40, 50, 60\}$ and $m = \{5, 10, 15, 20\}$. The large benchmark set, which is denoted by VRF_hard_large benchmark, consists of 240 large instances with combinations of $n = \{100, 200, 300, 400, 500, 600, 700, 800\}$ and $m = \{20, 40, 60\}$. In this paper, the optimal solutions, which is provided by Lin and Ying (2016), are employed to be the best makespan so far. All of the considered algorithms are tested with maximum running time $(n/2) \times m \times \rho$ *ms*, where $\rho = \{5, 15, 30\}$. Each algorithm runs 30 times on each instance. The following two subsections provide the computational results and statistical results of VRF_hard_small benchmark and VRF_hard_large benchmark respectively.

6.4.1. Results and comparisons for VRF_hard_small benchmark

The computational results for the VRF_hard_small benchmark are listed in Tables 10–12 where the best results are given in bold. As Tables 10 and 11, the average values of ARPD obtained by HBV for short running time are significantly better than the ARPD value obtained by compared algorithms. Besides, the SD value is also smaller than the value of SD obtained by other algorithms. From the Table 12, the average ARPD value obtained by HBV is 0.18,

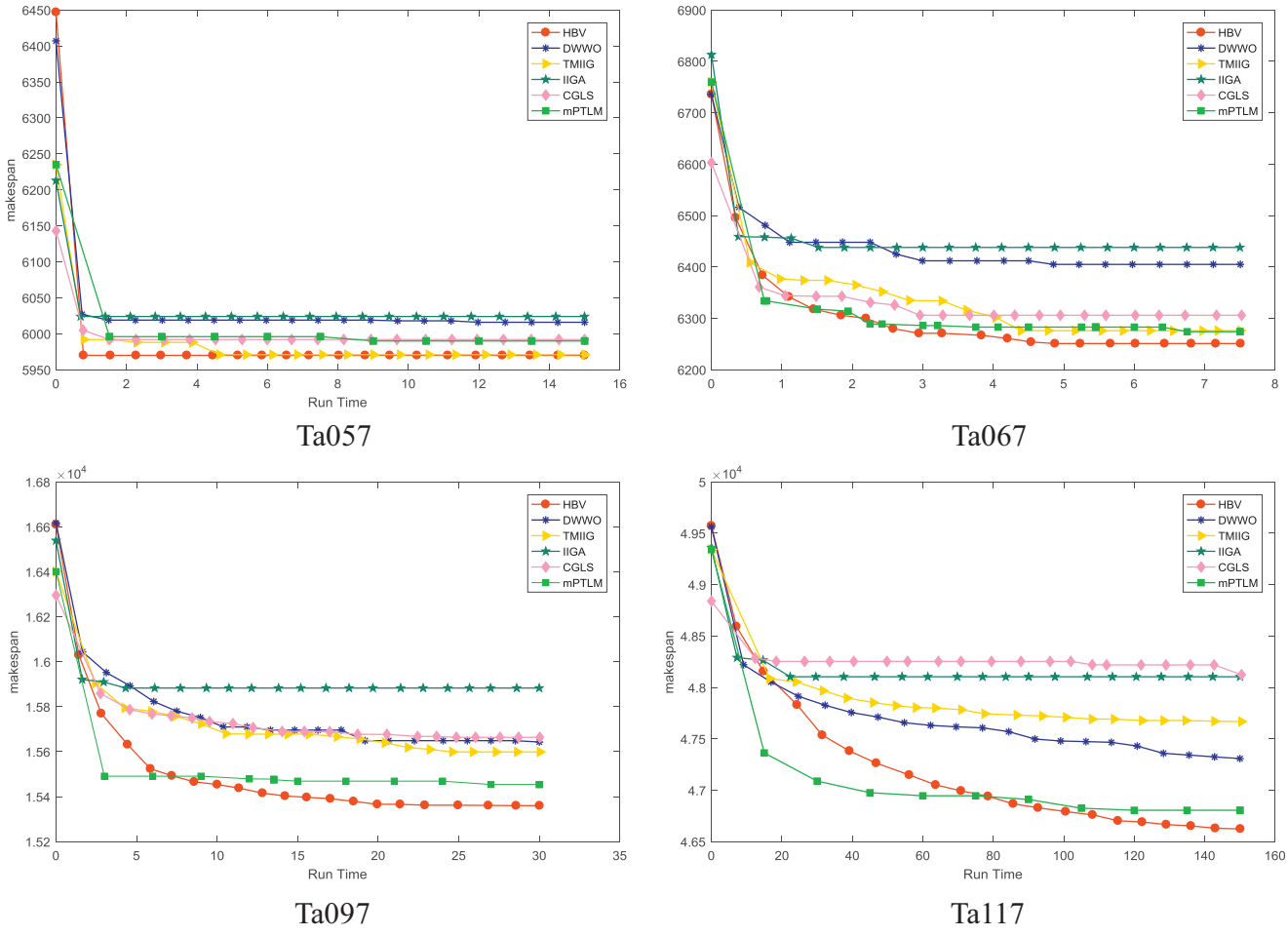

Fig. 5. Convergence curve for the instances of TA.

Table 10
Computational results of VRF_hard_small benchmark with $\rho = 5$.

Instance	$n \times m$	HBV		DWWO		mPTLM		TMIIG		IIGA		CGLS	
		ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD
10_5	10 × 5	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
10_10	10 × 10	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
10_15	10 × 15	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
10_20	10 × 20	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
20_5	20 × 5	**0.00**	**0.12**	0.10	1.51	0.03	0.28	0.01	0.29	0.05	0.50	0.17	1.44
20_10	20 × 10	0.00	0.04	0.03	1.02	0.01	0.23	0.01	0.29	**0.00**	**0.00**	0.07	1.39
20_15	20 × 15	**0.00**	**0.00**	0.01	0.33	0.00	**0.00**	0.01	0.62	0.00	0.13	0.07	0.53
20_20	20 × 20	**0.00**	**0.00**	0.03	1.03	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.05	1.14
30_5	30 × 5	**0.10**	2.37	0.44	4.83	0.15	**1.88**	0.25	2.81	0.70	3.67	0.66	5.97
30_10	30 × 10	**0.05**	**1.55**	0.32	5.86	0.13	2.82	0.14	3.60	0.17	2.61	0.45	6.80
30_15	30 × 15	0.04	**1.69**	0.17	4.59	0.05	2.45	**0.04**	2.55	0.04	2.24	0.21	7.69
30_20	30 × 20	**0.03**	**1.66**	0.20	5.90	0.07	3.84	0.07	3.15	0.08	2.37	0.30	8.34
40_5	40 × 5	**0.21**	3.70	0.71	7.06	0.25	**2.69**	0.55	5.41	1.26	5.73	0.89	8.12
40_10	40 × 10	**0.13**	**3.73**	0.47	7.40	0.19	4.29	0.35	5.69	0.48	4.66	0.81	11.62
40_15	40 × 15	**0.07**	**2.78**	0.29	7.36	0.13	4.27	0.19	6.17	0.22	4.48	0.70	13.12
40_20	40 × 20	**0.07**	**3.21**	0.28	9.76	0.15	6.13	0.20	7.51	0.24	5.66	0.41	9.00
50_5	50 × 5	**0.23**	4.36	0.81	7.66	0.37	**3.69**	0.74	7.65	1.83	8.51	1.33	14.36
50_10	50 × 10	**0.19**	5.59	0.63	10.63	0.24	**5.54**	0.47	8.10	0.85	7.67	0.99	15.31
50_15	50 × 15	**0.14**	**5.61**	0.44	10.31	0.25	7.57	0.39	8.66	0.67	7.17	0.71	12.78
50_20	50 × 20	**0.13**	**5.32**	0.45	8.41	0.30	10.19	0.36	10.95	0.58	7.56	0.74	19.24
60_5	60 × 5	**0.33**	5.96	1.18	9.24	0.59	**5.53**	1.08	11.17	2.26	7.16	1.53	15.54
60_10	60 × 10	0.28	7.33	0.72	10.57	**0.28**	**7.20**	0.84	12.19	1.24	9.39	1.50	18.40
60_15	60 × 15	**0.20**	**8.23**	0.69	11.77	0.42	9.95	0.56	14.03	1.04	11.35	1.39	22.76
60_20	60 × 20	**0.15**	**7.72**	0.56	12.45	0.42	13.33	0.52	15.56	0.80	9.89	1.20	19.72
Avg		**0.23**	**2.96**	0.49	5.74	0.30	3.83	0.42	5.27	0.66	4.20	0.73	8.89

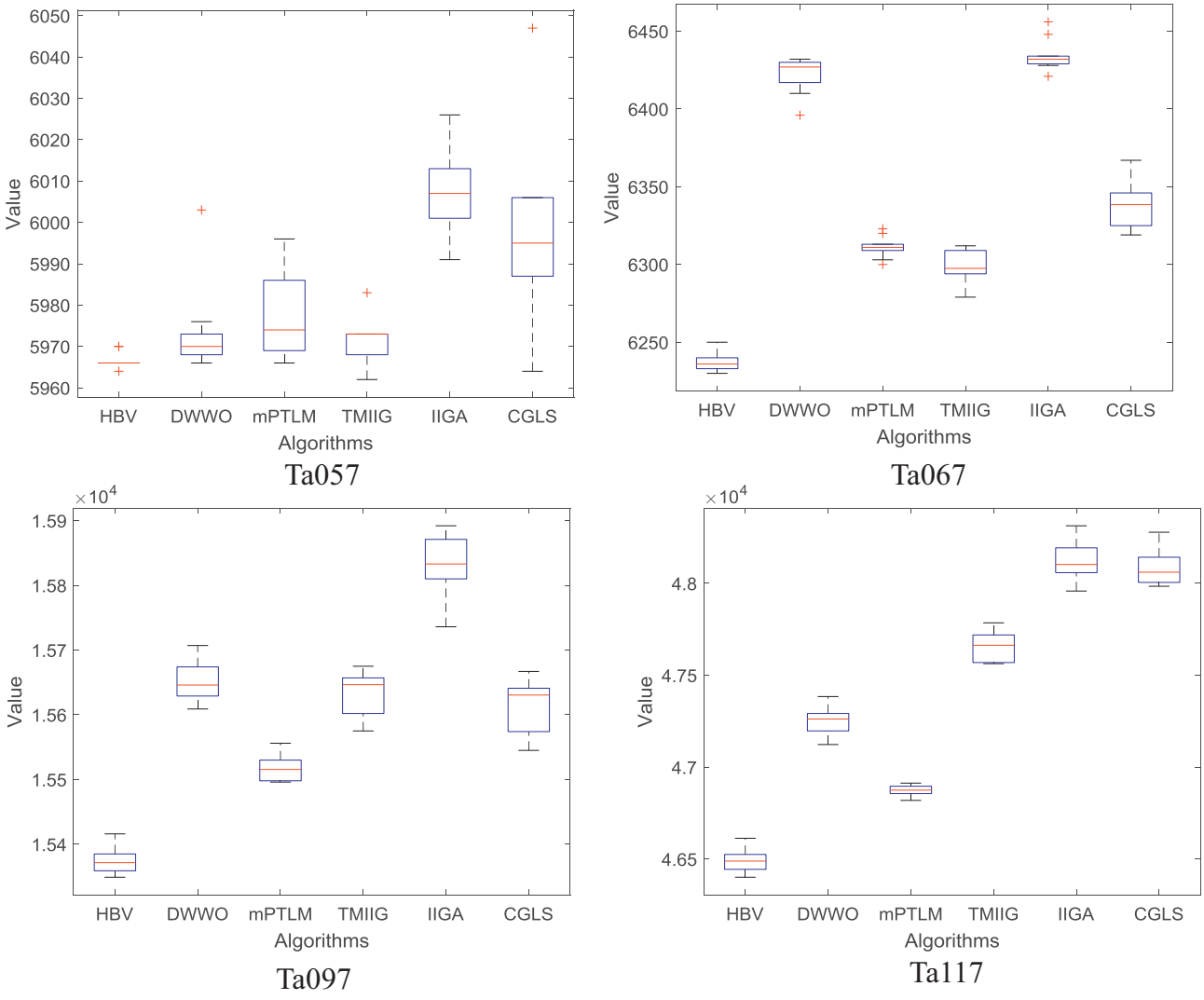

Fig. 6. Boxplots for the instances of TA.

which is less than 0.34, 0.23, 0.28, 0.60 obtained by the DWWO, mPTLM, TMIIG, IIGA and CGLS respectively. Fig. 8 shows the interval plot for the interaction between the algorithms and the maximum running time. From Fig. 7, the result obtained by HBV significantly better than the other compared algorithms.

The multiple-problem Wilcoxon's test and Friedman's test are also carried out to detect the significant differences between HBV and other algorithms. As the results of Wilcoxon's test which listed in Table 13, the HBV provides higher $R+$ values than $R-$ values in all cases. The result shows that the HBV is significantly better among the compared algorithms for solving NWFSP problems with $\alpha = 0.05$, $\alpha = 0.01$. The result of Friedman's test is listed in Table 14, the HBV algorithm is the best ranking among the six algorithms. From Fig. 8, the solutions obtained by HBV algorithm are significantly better than the solutions obtained by other compared algorithms with $\alpha = 0.05$ and $\alpha = 0.01$ on VRF_hard_small benchmark.

The values of SD obtained by the six algorithms are also provided in Tables 10–12 to show the robustness of the HBV algorithm. The SD values obtained by HBV are smaller than other algorithms in most instances with all the running times. For $\rho = 30$, the average value of SD 0.18 obtained by HBV is less than the 0.34, 0.23, 0.28, 0.60, 0.55 obtained by DWWO, mPTLM, TMIIG, IIGA and CGLS respectively. The computational results demonstrate that the HBV algorithm is robust on VRF_hard_small benchmark.

6.4.2. Results and comparisons for VRF_hard_large benchmark

The Tables 15–17 lists the computational results for the VRF_hard_large benchmark where the best results are given in bold. From Tables 15 and 16, the values of ARPD obtained by HBV are smaller than other algorithms on most of the instances with short running time. However, the ARPD value of HBV is larger than other algorithms in several instances. The reason is given as follow. As the size of instance increase, it is difficult to find the optimal solutions in short running time. From Table 17, the ARPD obtained by HBV are smaller than other compared algorithms on the most of instances. The average ARPD value 0.98 obtained by HBV is significantly smaller than the 2.11, 1.41, 2.44, 3.81 and 3.15 obtained by DWWO, mPTLM, TMIIG, IIGA respectively. The Fig. 9 shows the interval plot for the interaction between the algorithms and the maximum running time. From Fig. 9, the result obtained by HBV significantly better than the other compared algorithms.

The multiple-problem Wilcoxon's test and Friedman's test are also carried out to detect the significant differences between HBV and other compared algorithms. As the results of Wilcoxon's test

Table 11
Computational results of VRF_hard_small benchmark with $\rho = 15$.

Instance	$n \times m$	HBV		DWWO		mPTLM		TMIIG		IIGA		CGLS	
		ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD
10_5	10 × 5	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
10_10	10 × 10	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
10_15	10 × 15	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
10_20	10 × 20	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
20_5	20 × 5	**0.00**	**0.00**	0.03	0.72	0.02	0.23	0.00	0.03	0.04	0.46	0.11	1.00
20_10	20 × 10	**0.00**	**0.00**	0.01	0.39	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.04	1.45
20_15	20 × 15	**0.00**	**0.00**	0.01	0.09	0.00	**0.00**	0.00	0.05	0.01	0.14	0.06	0.18
20_20	20 × 20	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.03	1.11
30_5	30 × 5	**0.04**	**1.28**	0.26	3.64	0.08	1.35	0.11	1.94	0.51	3.40	0.49	5.12
30_10	30 × 10	**0.01**	**0.77**	0.18	4.08	0.07	1.87	0.09	1.58	0.15	2.39	0.35	4.62
30_15	30 × 15	**0.00**	**0.24**	0.12	3.52	0.02	0.96	0.03	1.41	0.04	1.84	0.20	5.68
30_20	30 × 20	**0.02**	**0.74**	0.09	3.68	0.03	1.87	0.05	2.39	0.06	2.36	0.23	6.80
40_5	40 × 5	**0.12**	**2.27**	0.47	5.00	0.18	2.86	0.28	3.24	1.08	6.33	0.63	6.10
40_10	40 × 10	**0.08**	**2.60**	0.38	7.59	0.13	3.69	0.22	5.14	0.44	5.92	0.63	9.84
40_15	40 × 15	**0.02**	**1.74**	0.19	7.00	0.08	2.95	0.11	3.53	0.21	4.76	0.49	11.62
40_20	40 × 20	**0.04**	**2.06**	0.22	7.80	0.12	4.91	0.11	4.00	0.22	6.31	0.32	8.18
50_5	50 × 5	**0.14**	**3.37**	0.57	6.51	0.28	3.73	0.48	5.35	1.61	7.08	1.01	9.68
50_10	50 × 10	**0.12**	3.82	0.40	7.08	0.17	**3.72**	0.33	6.57	0.76	7.38	0.76	11.31
50_15	50 × 15	**0.10**	**3.55**	0.35	9.83	0.17	5.16	0.27	7.35	0.63	6.96	0.57	9.97
50_20	50 × 20	**0.09**	**3.65**	0.37	6.71	0.27	8.99	0.18	6.52	0.53	9.04	0.65	17.18
60_5	60 × 5	**0.20**	**4.26**	0.76	8.18	0.49	5.34	0.72	7.28	2.10	8.90	1.16	12.39
60_10	60 × 10	**0.17**	**5.47**	0.55	10.45	0.24	6.16	0.55	9.39	1.22	9.84	1.15	13.72
60_15	60 × 15	**0.11**	**5.52**	0.60	11.16	0.33	9.14	0.42	8.25	1.03	10.63	1.14	20.32
60_20	60 × 20	**0.10**	**6.07**	0.44	10.94	0.37	12.91	0.35	9.37	0.81	13.08	0.97	15.25
Avg		**0.19**	**1.97**	0.38	4.77	0.26	3.16	0.31	3.47	0.61	4.45	0.59	7.15

Table 12
Computational results of VRF_hard_small benchmark with ρ=30.

Instance	$n \times m$	HBV		DWWO		mPTLM		TMIIG		IIGA		CGLS	
		ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD
10_5	10 × 5	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
10_10	10 × 10	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
10_15	10 × 15	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
10_20	10 × 20	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
20_5	20 × 5	**0.00**	**0.00**	0.02	0.41	0.01	0.17	0.00	0.03	0.04	0.32	0.12	1.33
20_10	20 × 10	**0.00**	**0.00**	0.00	0.05	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.02	0.58
20_15	20 × 15	**0.00**	**0.00**	0.00	0.09	0.00	0.03	0.00	0.03	0.01	0.14	0.06	0.22
20_20	20 × 20	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	0.05	1.22
30_5	30 × 5	**0.01**	**0.96**	0.08	2.91	0.03	1.45	0.05	1.39	0.19	3.89	0.38	5.09
30_10	30 × 10	**0.01**	**0.48**	0.11	3.80	0.05	1.86	0.05	1.16	0.15	2.98	0.35	4.27
30_15	30 × 15	0.00	0.33	0.07	2.25	**0.00**	**0.09**	0.02	1.18	0.04	1.31	0.14	4.69
30_20	30 × 20	**0.01**	**0.78**	0.08	2.34	0.02	1.94	0.04	1.84	0.05	1.85	0.21	6.99
40_5	40 × 5	**0.07**	**1.64**	0.35	4.48	0.13	2.08	0.23	3.53	1.00	5.71	0.61	6.02
40_10	40 × 10	**0.06**	**1.79**	0.34	6.44	0.11	3.12	0.21	4.22	0.43	4.92	0.55	9.09
40_15	40 × 15	**0.01**	**0.94**	0.15	5.16	0.05	2.27	0.09	2.71	0.22	5.12	0.45	12.16
40_20	40 × 20	**0.03**	**1.29**	0.15	5.20	0.08	3.59	0.10	4.22	0.23	5.33	0.29	7.40
50_5	50 × 5	**0.11**	2.59	0.44	5.09	0.21	**2.59**	0.38	4.71	1.57	7.88	0.88	9.45
50_10	50 × 10	**0.09**	3.95	0.36	7.03	0.12	**3.33**	0.25	4.85	0.81	6.99	0.77	9.26
50_15	50 × 15	**0.08**	**3.46**	0.29	8.45	0.15	6.28	0.21	6.91	0.62	9.01	0.56	10.05
50_20	50 × 20	**0.07**	**3.47**	0.29	9.21	0.19	8.20	0.18	6.37	0.55	8.98	0.55	13.29
60_5	60 × 5	**0.16**	**3.69**	0.64	7.01	0.39	4.08	0.56	6.59	2.03	8.90	1.08	12.43
60_10	60 × 10	**0.13**	5.07	0.51	9.39	0.18	**4.76**	0.42	8.61	1.21	9.76	0.99	14.42
60_15	60 × 15	**0.08**	**4.53**	0.51	12.07	0.25	8.28	0.34	7.91	1.01	11.58	1.10	19.69
60_20	60 × 20	**0.08**	**4.57**	0.38	9.74	0.32	10.30	0.32	9.94	0.78	11.44	0.88	15.51
Avg		**0.18**	**1.65**	0.34	4.21	0.23	2.68	0.28	3.18	0.60	4.42	0.55	6.80

Table 13
Results of the multiple-problem Wilcoxon's test at $\alpha = 0.05$ and $\alpha = 0.01$ significance level.

HBV vs.	$R+$	$R-$	Z	p-value	$\alpha = 0.05$	$\alpha = 0.01$
DWWO	12,393.5	9.5	−10.853	1.93E−27	Yes	Yes
mPTLM	8987	743	−8.67044	4.30E−18	Yes	Yes
TMIIG	10,344	96	−10.22	1.61E−24	Yes	Yes
IIGA	13,011.5	29.5	−10.9562	6.20E−28	Yes	Yes
CGLS	15,040.0	11.0	−11.3905	4.66E−30	Yes	Yes

listed in Table 18, the HBV provides higher $R+$ values than $R-$ values in all cases. The result shows that the HBV is significantly better among the compared algorithms for solving NWFSP problems with $\alpha = 0.05$, $\alpha = 0.01$. The result of Friedman's test is listed in Table 19, the HBV algorithm is the best ranking among the five algorithms. From Fig. 10, the solutions obtained by HBV algorithm are significantly better than the solutions obtained by other compared algorithms with $\alpha = 0.05$ and $\alpha = 0.01$ on VRF_hard_large benchmark. As the Table 17, the average SD value 54.64 obtained

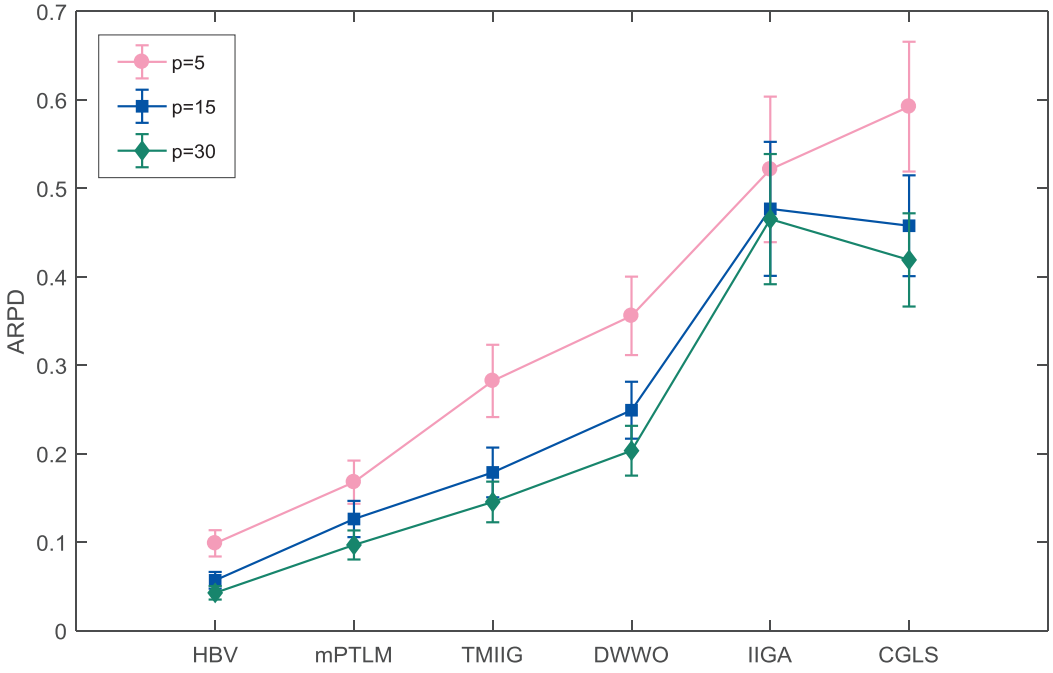

Fig. 7. The interval plot for the interaction between the algorithms and the maximum running time ρ (VRF_hard_small benchmark).

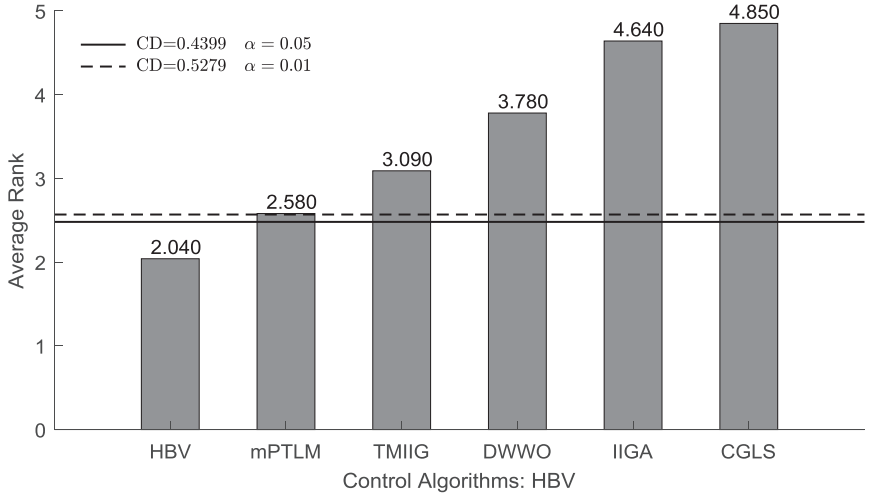

Fig. 8. Rankings obtained through the Friedman test and graphical representation.

Table 14
Ranking of algorithms obtained through Friedman's test.

Algorithm	Mean rank	Chi-Square	*p*-value
HBV	2.04	628.31	1.5419E−133
DWWO	3.78		
mPTLM	2.58		
TMIIG	3.09		
IIGA	4.64		
CGLS	4.85		
Crit. Diff. $\alpha = 0.05$	0.4399		
Crit. Diff. $\alpha = 0.01$	0.5279		

by HBV is smaller than the 69.48, 59.67, 69.55, 89.22 and 183.71 obtained by DWWO, mPTLM, TMIIG, IIGA and CGLS respectively. Therefore, the HBV is robust on VRF_hard_large benchmark.

6.5. Analysis of experimental results

As above experimental series, the proposed HBV algorithm is an effective and robust algorithm for solving the NWFSP with makespan criterion. The reasons are concluded as follows.

Firstly, the BBO algorithm, which is a classical population-based algorithm, has been proven to be an effective framework in various fields. Although the population-based algorithms are sensitive to the parameters of algorithms such as the population size, mutation probability, they in general provide excellent results on large-scale problems. In this paper, the parameters combinations of HBV have been analyzed by experiment. From the above experimental results, the population-based algorithms HBV, DWWO and mPTLM significantly outperform the TMIIG, IIGA and CGLS on the VRF_hard_large benchmark.

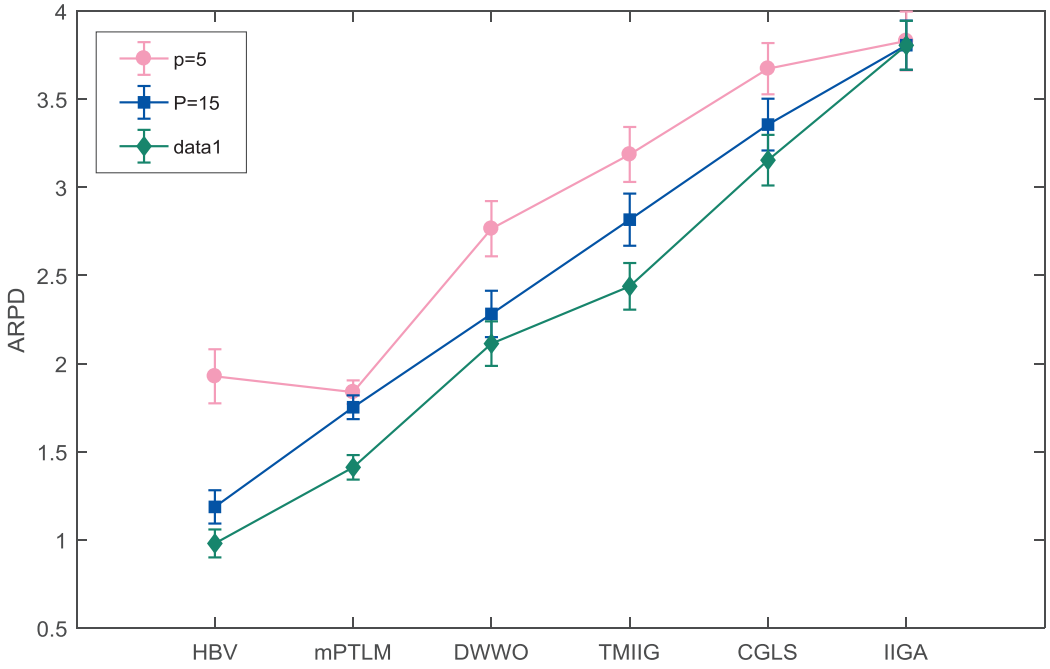

Fig. 9. The interval plot for the interaction between the algorithms and the maximum running time ρ (VRF_hard_large benchmark).

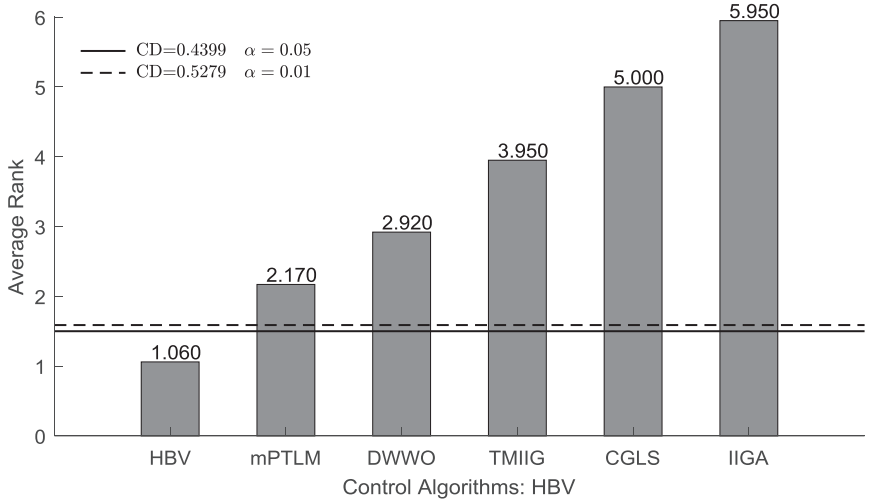

Fig. 10. Rankings obtained through the Friedman test and graphical representation.

Secondly, the performance of algorithm is affected by the different coding scheme. For mPTLM, the LOV scheme extends the search space of NWFSP. However, the proposed HBV algorithm only searches in the job-permutation search space. Therefore, the search efficiency of HBV outperforms the search efficiency of mPTLM.

Thirdly, the neighborhood search is a significant factor which affects the performance of algorithms. In the existing researches, the insert and swap are the most common neighborhood structure for solving the NWFSP. However, the different neighborhood structures have a gap in various instances. The block is also an effective neighborhood structure. In the HBV algorithm, the block neighborhood structure is employed to construct the self-improvement strategy and the variable neighbor search.

Finally, although the CGLS is an excellent algorithm for the mixed blocking permutation flow shop scheduling problem, the proposed HBV outperforms the CGLS as the no-free-launch theorem.

It is worth noting that the HBV seems to be similar to the DWWO which was proposed by us earlier. However, the difference between HBV and DWWO has been proven by the above experimental results. For all the benchmarks, the performance of HBV significantly outperforms the performance of DWWO. The main difference between the two algorithms is described as follows. Firstly, the optimization process of HBV mainly depends on the interaction of individuals in the population. In this paper, the path relink technique is introduced to perform the migration operator of HBV. In DWWO, the individuals search in parallel but rarely share information. Therefore, the HBV has faster convergence speed than DWWO. Secondly, the block neighborhood structure is employed by HBV as the excellent solutions have certain similarities. The

Table 15
Computational results of VRF_hard_large benchmark with $\rho = 5$.

Instance	$n \times m$	HBV		DWWO		mPTLM		TMIIG		IIGA		CGLS	
		ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD
100_20_1	100 × 20	**0.34**	**14.92**	0.84	16.06	1.02	31.87	1.21	27.40	1.78	18.24	1.78	38.90
100_40_1	100 × 40	**0.33**	**17.58**	0.67	20.08	1.10	48.70	0.96	26.77	1.53	24.87	1.43	43.91
100_60_1	100 × 60	**0.29**	**20.75**	0.66	29.14	1.38	48.94	0.90	32.98	1.68	29.29	1.63	60.02
200_20_1	200 × 20	**0.90**	30.28	1.94	38.52	1.04	**28.80**	2.38	39.83	2.94	40.41	2.87	74.36
200_40_1	200 × 40	**0.73**	**32.74**	1.43	34.22	1.68	87.43	1.89	48.17	2.70	51.24	2.37	88.50
200_60_1	200 × 60	**0.63**	**37.16**	1.30	42.52	1.85	163.80	1.75	60.05	2.68	63.12	2.63	133.75
300_20_1	300 × 20	1.52	43.12	2.69	50.32	**1.09**	**38.26**	3.20	49.24	3.66	48.50	3.70	86.49
300_40_1	300 × 40	**1.05**	**46.77**	2.03	58.14	1.86	99.78	2.75	67.44	3.42	70.30	3.29	119.56
300_60_1	300 × 60	**0.98**	**51.79**	1.93	76.94	2.36	196.68	2.38	69.91	3.38	82.95	2.89	187.25
400_20_1	400 × 20	2.22	59.60	3.24	61.74	**1.25**	**33.69**	3.76	56.44	4.09	65.90	4.38	89.43
400_40_1	400 × 40	**1.47**	**66.84**	2.57	71.05	1.72	120.86	3.00	73.88	3.82	79.03	3.52	137.70
400_60_1	400 × 60	**1.24**	**77.55**	2.39	91.89	2.49	189.55	2.75	93.63	3.82	108.97	3.84	193.42
500_20_1	500 × 20	2.77	79.46	3.74	67.86	**1.56**	**49.44**	4.20	54.61	4.55	75.18	4.50	79.77
500_40_1	500 × 40	1.94	85.05	2.98	94.37	**1.80**	114.48	3.34	**68.90**	4.23	107.76	4.17	168.80
500_60_1	500 × 60	**1.47**	**91.98**	2.65	110.78	2.53	270.10	3.15	102.69	4.12	120.03	3.66	205.13
600_20_1	600 × 20	3.28	82.57	4.20	83.15	**1.74**	**52.79**	4.54	67.34	4.85	98.65	4.95	67.32
600_40_1	600 × 40	2.31	90.71	3.27	**85.53**	**1.90**	99.00	3.76	87.34	4.44	125.88	4.44	152.12
600_60_1	600 × 60	**2.01**	**95.98**	3.05	117.98	2.55	214.92	3.65	98.30	4.38	128.58	3.96	189.24
700_20_1	700 × 20	4.13	87.54	4.80	76.03	**1.89**	**57.31**	5.02	68.75	5.28	100.11	5.24	64.58
700_40_1	700 × 40	3.05	105.69	3.69	104.56	**1.97**	106.76	4.01	**97.55**	4.73	119.49	4.52	126.89
700_60_1	700 × 60	**2.53**	117.57	3.38	133.14	2.61	189.00	3.85	**106.62**	4.52	151.58	4.24	171.68
800_20_1	800 × 20	4.59	93.59	5.20	71.20	**2.05**	68.41	5.38	**56.96**	5.70	106.23	5.29	76.65
800_40_1	800 × 40	3.48	105.88	3.98	113.79	**2.08**	110.21	4.50	**95.85**	4.84	111.65	4.53	133.92
800_60_1	800 × 60	3.01	127.25	3.73	129.18	**2.62**	162.02	4.15	**98.89**	4.77	157.65	4.33	171.19
Avg		1.93	69.27	2.77	74.09	**1.84**	107.62	3.19	**68.73**	3.83	86.90	3.67	119.19

Table 16
Computational results of VRF_hard_large benchmark with $\rho = 15$.

Instance	$n \times m$	HBV		DWWO		mPTLM		TMIIG		IIGA		CGLS	
		ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD
100_20_1	100 × 20	**0.24**	**10.42**	0.67	13.15	0.90	27.33	0.89	20.19	1.72	21.87	1.48	33.08
100_40_1	100 × 40	**0.24**	**17.27**	0.56	20.90	0.95	36.45	0.75	23.43	1.52	31.51	1.18	45.55
100_60_1	100 × 60	**0.20**	**19.50**	0.55	25.68	1.21	48.18	0.71	28.21	1.65	34.47	1.51	61.57
200_20_1	200 × 20	**0.60**	**23.35**	1.42	29.69	0.88	27.77	1.91	34.13	2.94	35.08	2.40	76.17
200_40_1	200 × 40	**0.52**	**26.02**	1.10	33.20	1.51	92.38	1.51	41.47	2.73	55.29	2.02	85.14
200_60_1	200 × 60	**0.45**	**40.79**	1.04	45.86	1.68	146.66	1.41	52.14	2.66	65.62	2.39	117.01
300_20_1	300 × 20	**0.85**	36.64	1.98	42.83	1.07	**27.87**	2.68	50.94	3.63	53.23	3.42	118.30
300_40_1	300 × 40	**0.70**	**45.73**	1.67	55.78	1.62	72.01	2.35	64.59	3.38	70.28	2.95	125.40
300_60_1	300 × 60	**0.69**	**57.18**	1.53	67.24	1.99	170.71	2.06	61.90	3.34	80.77	2.59	194.05
400_20_1	400 × 20	**1.19**	41.21	2.56	57.89	1.25	**35.54**	3.38	61.07	4.11	66.02	3.86	143.79
400_40_1	400 × 40	**0.92**	**48.65**	2.10	65.61	1.62	78.87	2.73	66.64	3.84	82.98	3.07	179.23
400_60_1	400 × 60	**0.86**	**71.41**	2.05	83.56	2.31	159.91	2.51	90.43	3.82	105.94	3.40	249.32
500_20_1	500 × 20	**1.53**	50.05	3.02	67.75	1.53	**48.48**	3.88	73.89	4.55	90.95	4.10	108.45
500_40_1	500 × 40	**1.13**	**72.96**	2.46	78.91	1.77	91.12	3.09	75.56	4.24	101.23	3.94	219.23
500_60_1	500 × 60	**0.98**	**72.60**	2.32	97.78	2.36	167.55	2.93	102.52	4.10	126.05	3.27	234.58
600_20_1	600 × 20	1.99	55.86	3.47	73.21	**1.74**	**55.00**	4.22	69.07	4.88	94.25	4.67	120.24
600_40_1	600 × 40	**1.39**	**69.98**	2.84	82.84	1.90	81.63	3.42	97.03	4.46	104.38	4.19	190.16
600_60_1	600 × 60	**1.21**	**78.58**	2.66	104.91	2.51	174.24	3.23	126.48	4.37	135.58	3.60	314.25
700_20_1	700 × 20	2.65	81.76	3.88	83.99	**1.89**	**46.07**	4.59	81.72	5.19	82.19	4.97	127.17
700_40_1	700 × 40	**1.71**	88.23	3.08	105.04	1.98	**87.43**	3.62	96.55	4.70	123.88	4.23	194.99
700_60_1	700 × 60	**1.50**	**108.99**	2.99	119.06	2.58	169.20	3.41	133.07	4.53	159.96	3.99	245.82
800_20_1	800 × 20	3.11	76.72	4.22	82.40	**2.06**	**60.21**	4.81	79.90	5.39	123.09	5.06	120.24
800_40_1	800 × 40	**2.06**	91.55	3.34	113.69	2.07	**90.50**	3.92	99.74	4.81	128.31	4.19	210.19
800_60_1	800 × 60	**1.75**	**113.41**	3.22	135.90	2.66	171.56	3.59	126.64	4.79	138.27	4.06	254.91
Avg		**1.19**	**58.29**	2.28	70.29	1.75	90.28	2.82	73.22	3.81	87.97	3.36	157.03

experimental results show that the self-improvement strategy which is based on the block neighborhood structure remarkably improves the performance of HBV. Thirdly, the IG algorithm was embedded into the mutation operator to extend the exploitation ability of HBV. In DWWO, the IG was applied to the propagation operator to search the promising area adaptively. Finally, the VNS is different in HBV and DWWO as the neighbor structure is different.

7. Conclusion and future research

In this paper, a hybrid biogeography-based optimization with variable neighborhood search (HBV) is presented to solve the no-wait flow shop scheduling problem (NWFSP) with the objective of minimizing makespan. The HBV includes five phases: Firstly, a hybrid initial population strategy based on the NN+MNEH is employed to generate potential solutions. Secondly, the migration

Table 17
Computational results of VRF_hard_large benchmark with ρ=30.

Instance	$n \times m$	HBV		DWWO		mPTLM		TMIIG		IIGA		CGLS	
		ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD	ARPD	SD
100_20_1	100 × 20	**0.18**	**8.92**	0.58	13.95	0.39	13.90	0.74	19.34	1.77	20.49	1.39	37.25
100_40_1	100 × 40	**0.20**	**14.93**	0.49	17.19	0.64	39.66	0.59	19.45	1.56	24.11	1.13	38.59
100_60_1	100 × 60	**0.15**	**15.39**	0.51	24.15	0.86	48.08	0.61	23.18	1.62	32.50	1.32	51.36
200_20_1	200 × 20	**0.50**	26.26	1.21	29.60	0.76	**15.34**	1.55	33.37	2.97	36.69	2.26	65.75
200_40_1	200 × 40	**0.44**	**30.80**	1.00	33.59	0.81	38.02	1.26	39.38	2.71	49.80	1.95	82.04
200_60_1	200 × 60	**0.37**	**30.20**	0.95	43.82	1.30	95.27	1.21	54.59	2.65	65.37	2.11	105.56
300_20_1	300 × 20	**0.68**	27.82	1.77	42.07	1.23	**20.93**	2.31	39.08	3.63	61.25	3.20	120.00
300_40_1	300 × 40	**0.62**	39.85	1.50	56.44	0.90	**35.70**	1.95	61.88	3.37	86.05	2.70	173.70
300_60_1	300 × 60	**0.60**	**42.34**	1.44	62.22	1.23	88.14	1.72	73.13	3.34	87.65	2.45	196.00
400_20_1	400 × 20	**0.93**	35.08	2.27	63.16	1.56	**30.61**	2.81	52.35	4.11	55.70	3.49	158.48
400_40_1	400 × 40	**0.73**	49.84	1.89	62.87	1.08	**43.02**	2.41	66.27	3.84	78.29	2.83	211.71
400_60_1	400 × 60	**0.70**	**60.02**	1.83	76.96	1.28	76.62	2.23	83.45	3.84	110.81	3.17	236.67
500_20_1	500 × 20	**1.16**	40.07	2.72	57.66	1.87	**33.21**	3.43	56.93	4.56	82.70	3.75	149.07
500_40_1	500 × 40	**1.02**	**58.52**	2.37	76.53	1.25	59.16	2.71	83.65	4.22	103.11	3.75	243.32
500_60_1	500 × 60	**0.87**	83.62	2.17	94.81	1.39	**76.35**	2.46	104.78	4.13	115.94	3.01	264.30
600_20_1	600 × 20	**1.37**	48.96	3.03	70.58	2.17	**43.29**	3.64	69.00	4.84	96.58	4.44	171.91
600_40_1	600 × 40	**1.11**	82.55	2.63	80.37	1.45	**62.74**	2.92	110.90	4.49	108.36	3.99	257.87
600_60_1	600 × 60	**1.04**	**88.02**	2.53	101.62	1.70	90.11	2.72	99.27	4.34	150.97	3.48	333.92
700_20_1	700 × 20	**2.06**	61.10	3.62	81.97	2.37	**47.63**	3.96	67.11	5.15	101.44	4.73	138.77
700_40_1	700 × 40	**1.33**	**71.32**	2.90	100.49	1.61	79.83	3.16	85.41	4.70	119.06	4.07	257.81
700_60_1	700 × 60	**1.25**	**98.90**	2.84	130.95	1.72	133.16	3.03	117.04	4.53	134.21	3.80	324.68
800_20_1	800 × 20	2.74	104.07	4.03	96.97	**2.57**	**58.99**	4.23	85.78	5.43	110.00	4.77	162.04
800_40_1	800 × 40	1.84	95.30	3.27	111.37	**1.78**	**78.34**	3.49	101.45	4.80	126.77	4.02	309.19
800_60_1	800 × 60	**1.61**	**97.51**	3.18	138.21	1.94	123.87	3.39	122.42	4.75	183.57	3.90	319.01
Avg		**0.98**	**54.64**	2.11	69.48	1.41	59.67	2.44	69.55	3.81	89.22	3.15	183.71

Table 18
Results of the multiple-problem Wilcoxon's test at α = 0.05 and α = 0.01 significance level.

HBV vs.	$R+$	$R-$	Z	p-value	$\alpha = 0.05$	$\alpha = 0.01$
DWWO	28,920.00	0.00	−13.43	4.00E−41	Yes	Yes
mPTLM	28,125.50	554.50	−12.88	5.49E−38	Yes	Yes
TMIIG	28,920.00	0.00	−13.43	4.00E−41	Yes	Yes
IIGA	28,920.00	0.00	−13.43	4.00E−41	Yes	Yes
CGLS	28,920.00	0.00	−13.43	4.00E−41	Yes	Yes

Table 19
Ranking of algorithms obtained through Friedman's test.

Algorithm	Mean rank	Chi-Square	p-value
HBV	1.06	1117.56	2.108E−239
DWWO	2.92		
mPTLM	2.17		
TMIIG	3.95		
IIGA	5.90		
CGLS	5.00		
Crit. Diff. $\alpha = 0.05$	0.4399		
Crit. Diff. α=0.01	0.5279		

operator is employed to guide the population to search the better area and the self-improvement strategy is employed to search the neighbor of unmigrated solutions. Thirdly, the mutation operator based on IG is used to maintain the diversity of the population. Furthermore, the improved variable search is utilized to enhance the quality of global best solutions in each generation. Finally, the elitism strategy is used to reserve the best solution in the current population. The computational results based on Taillard's and VRF benchmark show the effectiveness of HBV for solving the NWFSP.

Further research will be conducted in following directions. Firstly, the HBV algorithm will further extend to the NWFSP involving additional realistic conditions. Secondly, it is necessary to design novel and effective neighborhood structures. Thirdly, it is desirable to apply the HBV to other combinational optimization problems, such as traveling salesman problem, job shop scheduling problem, etc.

Credit authorship contribution statement

Fuqing Zhao: Funding acquisition, Investigation, Supervision, Writing - review & editing. **Shuo Qin:** Investigation, Software, Writing - original draft. **Yi Zhang:** Conceptualization, Formal analysis. **Weimin Ma:** Methodology, Resources. **Chuck Zhang:** Project administration, Writing - review & editing. **Houbin Song:** Visualization.

Acknowledgments

This work was financially supported by the National Natural Science Foundation of China under grant numbers 61663023. It was also supported by the Key Research Programs of Science and Technology Commission Foundation of Gansu Province (2017GS10817), Lanzhou Science Bureau project (2018-rc-98), Public Welfare Project of Zhejiang Natural Science Foundation(LGJ19E050001), Wenzhou Public Welfare Science and Technology project (G20170016), respectively.

References

Aljarah, I., Faris, H., Mirjalili, S, & Al-Madi, N. J. N. C. (2018). Training radial basis function networks using biogeography-based optimizer. *Neural Computing & Applications, 29*, 529–553.
Bonney, M. C., & Gundry, S. W. (1976). Solutions to the constrained flowshop sequencing problem. *Journal of the Operational Research Society, 27*, 869–883.
Burton and Letters. (1985). Pointwise properties of convergence in probability. *Statistics & Probability Letters, 3*, 315–316.
Clerc, & Kennedy (2002). The particle swarm – Explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation, 6*, 58–73.
Ding, J. Y., Song, S., Gupta, J. N. D., Rui, Z., Chiong, R., & Cheng, W. (2015a). An improved iterated greedy algorithm with a Tabu-based reconstruction strategy for the no-wait flowshop scheduling problem. *Applied Soft Computing, 30*, 604–613.
Ding, J. Y., Song, S., Gupta, J. N. D., Rui, Z., Chiong, R., & Cheng, W. (2015b). *A novel block-shifting simulated annealing algorithm for the no-wait flowshop scheduling problem*. New York: IEEE.
Ferrer, A., Guimarans, D., Ramalhinho, H., & Juan, A. A (2016). A BRILS metaheuristic for non-smooth flow-shop problems with failure-risk costs. *Expert Systems with Applications, 44*, 177–186.
Framinan, & Leisten (2008). Total tardiness minimization in permutation flow shops: A simple approach based on a variable greedy algorithm. *International Journal of Production Research, 46*, 6479–6498.

Gao, K., Xie, S., Hua, J., & Li, J. (2012). *A composite heuristic for the no-wait flow shop scheduling*. New York:: IEEE.
Gao, K., Xie, S., Hua, J., & Li, J. (2011). Discrete harmony search algorithm for the no wait flow shop scheduling problem with makespan criterion. In D. S. Huang, Y. Gan, V. Bevilacqua, & J. C. Figueroa (Eds.). In *Advanced intelligent computing: 6838* (pp. 592–599).
Garey, & Johnson (1979). *Computers and intractability: A guide to the theory of NP–Completeness*. W. H. Freeman.
Geem, Z. W., Kim, J. H., & Loganathan, G. (2001). A new heuristic optimization algorithm: Harmony search. *Simulation, 76*, 60–68.
Hansen, & Mladenović (1997). *Variable neighborhood search*. Elsevier Science Ltd.
Kononova, & Kochetov (2013). The variable neighborhood search for the two machine flow shop problem with a passive prefetch. *Journal of Applied and Industrial Mathematics, 7*, 54–67.
Lei, D. (2015). Variable neighborhood search for two-agent flow shop scheduling problem. *Computers & Industrial Engineering, 80*, 125–131.
Lei, D., Liang, G., & Zheng, Y (2018). A novel teaching-learning-based optimization algorithm for energy-efficient scheduling in hybrid flow shop. *IEEE Transactions on Engineering Management, 65*, 330–340.
Lei, & Guo (2011). Variable neighbourhood search for minimising tardiness objectives on flow shop with batch processing machines. *International Journal of Production Research, 49*, 519–529.
Lin, J. (2015). A hybrid biogeography-based optimization for the fuzzy flexible job-shop scheduling problem. *Knowledge-Based Systems, 78*, 59–74.
Lin, J. (2016). A hybrid discrete biogeography-based optimization for the permutation flowshop scheduling problem. *International Journal of Production Research, 54*, 4805–4814.
Lin, S. W., & Ying, K. C. (2016). Optimization of makespan for no-wait flowshop scheduling problems using efficient matheuristics. *Omega, 64*, 115–125.
Liu, S., Wang, P., & Zhang, J. (2018). An improved biogeography-based optimization algorithm for blocking flow shop scheduling problem. *Chinese Journal of Electronics, 27*, 351–358.
Ma (2010). An analysis of the equilibrium of migration models for biogeography-based optimization. *Information Sciences, 180*, 3444–3464.
Ma, Dan, S, & Fei, M. (2014). On the convergence of biogeography-based optimization for binary problems. *Mathematical Problems in Engineering, 11*, 1–11.
Montgomery (1976). *Design and analysis of experiments*. Wiley.
Moslehi, & Khorasanian (2014). A hybrid variable neighborhood search algorithm for solving the limited-buffer permutation flow shop scheduling problem with the makespan criterion. *Computers & Operations Research, 52*, 260–268.
Nawaz, M., Jr, & Ham, I.EEE.. (1983). A heuristic algorithm for the m -machine, n -job flow-shop sequencing problem. *Omega, 11*, 91–95.
Pan, Q. K., Tasgetiren, M. F., & Liang, Y. C. (2008a). A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem. *Computers & Operations Research, 35*, 2807–2839.
Pan, Q. K., Wang, L., & Zhao, B. H. (2008b). An improved iterated greedy algorithm for the no-wait flow shop scheduling problem with makespan criterion. *International Journal of Advanced Manufacturing Technology, 38*, 778–786.
Perez-Gonzalez, & Framinan (2010). Setting a common due date in a constrained flowshop: A variable neighbourhood search approach. *Computers & Operations Research, 37*, 1740–1748.
Qu, C., Fu, Y., Yi, Z., & Tan, J. (2018). Solutions to no-wait flow shop scheduling problem using the flower pollination algorithm based on the hormone modulation mechanism. *Complexity, 18*, 1–18.
Rabiee, M., Jolai, F., Asefi, H., Fattahi, P., & Lim, S. (2016). A biogeography-based optimisation algorithm for a realistic no-wait hybrid flow shop with unrelated parallel machines to minimise mean tardiness. *International Journal of Computer Integrated Manufacturing, 29*, 1007–1024.
Rajendran (1994). A no-wait flowshop scheduling heuristic to minimize makespan. *Journal of the Operational Research Society, 45*, 472–478.
Rashedi, E., Nezamabadi-Pour, H., & Saryazdi, S. (2009). GSA: A gravitational search algorithm. *Information Sciences, 179*, 2232–2248.
Riahi, V., Khorramizadeh, M., Newton, M. A. H., & Sattar, A. (2017). Scatter search for mixed blocking flowshop scheduling. *Expert Systems with Applications, 79*, 20–32.
Riahi, V., Newton, M. A. H., Su, K., & Sattar, A. (2019). Constraint guided accelerated search for mixed blocking permutation flowshop scheduling. *Computers & Operations Research, 102*, 102–120.
Ribas, & Companys (2013). A competitive variable neighbourhood search algorithm for the blocking flow shop problem. *European Journal of Industrial Engineering, 7*, 729–754.
Ribas, I., Companys, R., & Tort-Martorell, X. (2017). Efficient heuristics for the parallel blocking flow shop scheduling problem. *Expert Systems with Applications, 74*, 41–54.
Shao, W., Pi, D., & Shao, Z. (2017a). An extended teaching-learning based optimization algorithm for solving no-wait flow shop scheduling problem. *Applied Soft Computing, 61*, 193–210.
Shao, W., Pi, D., & Shao, Z. (2017b). Optimization of makespan for the distributed no-wait flow shop scheduling problem with iterated greedy algorithms. *Knowledge-Based Systems, 137*, 163–181.
Shao, W., Pi, D., & Shao, Z. (2018). A hybrid discrete teaching-learning based meta-heuristic for solving no-idle flow shop scheduling problem with total tardiness criterion. *Computers & Operations Research, 94*, 89–105.
Simon (2008). Biogeography-based optimization. *IEEE Transactions on Evolutionary Computation, 12*, 702–713.
Taillard (1993). Benchmarks for basic scheduling problems. *European Journal of Operational Research, 64*, 278–285.
Tasgetiren, M. F., Buyukdagli, O., Pan, Q. K., & Suganthan, P. N. (2013). A general variable neighborhood search algorithm for the no-idle permutation flowshop scheduling problem. In B. K. Panigrahi, P. N. Suganthan, S. Das, & S. S. Dash (Eds.). *Swarm, evolutionary, and memetic computing, Pt I*: 8297. Berlin: Springer-Verlag Berlin pp. 24-34.
Vallada, E., Ruiz, R., & Framinan, J. M. (2015). New hard benchmark for flowshop scheduling problems minimising makespan. *European Journal of Operational Research, 240*, 666–677.
Van Laarhoven, Aarts, E. H. L., & Lenstra (1992). Job shop scheduling by simulated annealing. *Operations Research, 40*, 113–125.
Wang, F., Deng, G., Jiang, T., & Zhang, S. (2018). Multi-objective parallel variable neighborhood search for energy consumption scheduling in blocking flow shops. *IEEE Access, 6*, 68686–68700.
Wang, X. H., & Duan (2014). A hybrid biogeography-based optimization algorithm for job shop scheduling problem. *Computers & Industrial Engineering, 73*, 96–114.
Wang, C., Li, X., & Wang, Q. (2010). Accelerated tabu search for no-wait flowshop scheduling problem with maximum lateness criterion. *European Journal of Operational Research, 206*, 64–72.
Xie, Z. P., Zhang, C. Y., Shao, X. Y., & Yin, Y (2014). Minimizing total flow time in a flow shop with blocking using a hybrid variable neighborhood search and simulated annealing algorithm. In *Applied mechanics and materials: 631* (pp. 57–61). Trans Tech Publ.
Ye, H., Li, W., & Abedini, A. (2017). An improved heuristic for no-wait flow shop to minimize makespan. *Journal of Manufacturing Systems, 44*, 273–279.
Ye, H., Wei, L., & Miao, E. (2016). An effective heuristic for no-wait flow shop production to minimize makespan. *Journal of Manufacturing Systems, 40*, 2–7.
Yogesh, C. K., Hariharan, M., Ngadiran, R., Adom, A. H., Yaacob, S., Berkai, C., & Polat, K. (2017). A new hybrid PSO assisted biogeography-based optimization for emotion and stress recognition from speech signal. *Expert Systems with Applications, 69*, 149–158.
Zhao, F., Liu, Y., Zhang, Y., Ma, W., & Zhang, C. (2017). A hybrid harmony search algorithm with efficient job sequence scheme and variable neighborhood search for the permutation flow shop scheduling problems. *Engineering Applications of Artificial Intelligence, 65*, 178–199.
Zhao, F., Liu, H., Zhang, Y., Ma, W., & Zhang, C. (2018). A discrete water wave optimization algorithm for no-wait flow shop scheduling problem. *Expert Systems with Applications, 91*, 347–363.
Zhao, F., Qin, S., Zhang, Y., Ma, W., Zhang, C., & Song, H. (2019). A two-stage differential biogeography-based optimization algorithm and its performance analysis. *Expert Systems with Applications, 115*, 329–345.
Zheng (2015). Water wave optimization: A new nature-inspired metaheuristic. *Computers & Operations Research, 55*, 1–11.
Zhou (2012). Optimization of flow shop scheduling problem using differential evolution and variable neighborhood search. In *Advanced materials research: 590* (pp. 540–544). Trans Tech Publ.