International Journal of Production Economics 193 (2017) 21–30

Contents lists available at ScienceDirect

International Journal of Production Economics

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

A new improved NEH heuristic for permutation flowshop scheduling problems

Weibo Liu, Yan Jin [*], Mark Price

School of Mechanical and Aerospace Engineering, Queen's University Belfast, Ashby Building, Belfast, BT9 5AH, UK

ARTICLE INFO

Keywords:
Scheduling
Heuristic
Flowshop
Makespan

ABSTRACT

Job evaluation and differentiation are crucial in scheduling. Since jobs can be represented by vectors of processing times, the average, standard deviation, and skewness of job processing times can be defined as the moments of their probability distribution. The first and the second moments of processing times are effective in sorting jobs (Dong et al., 2008), however they are not yet optimized to characterize and differentiate distributions of similar jobs. In this paper, skewness is utilized for the first time to construct a new priority rule, which is applied to the Nawaz-Enscore-Ham (NEH) heuristic (Nawaz et al., 1983), for solving scheduling problems in permutation flowshops. A novel tie-breaking rule is also developed by minimizing partial system idle time without increasing computational complexity of the NEH heuristic, in order to further improve the heuristic performance. Computational results show that the new heuristic outperforms the best NEH-based heuristics reported in the literature in terms of solution quality.

Notations

n	Total number of jobs
m	Total number of machines
J	Total number of instances
P	Total number of referenced heuristics
i	Index for job, $1 \le i \le n$
k	Index for machine, $1 \le k \le m$
j	Index for instance, $1 \le j \le J$
p	Index for heuristic, $1 \le p \le P$
$[i]$	The *i*th job of schedule
$t_{i,k}$	Processing time of job i on machine k
$f_{[i],k}$	The earliest relative completion time of the *i*th job on machine k
$q_{[i],k}$	The tail time of the *i*th job on machine k
$C_{[i],k}$	Completion time of the ith job on machine k
AVG_i	The average processing time of job i
STD_i	The standard deviation of processing times of job i
SKE_i	The skewness of processing times of job i

1. Introduction

In scheduling, job differentiation and sequencing have direct impacts on heuristic performance. Many successful scheduling rules and algorithms have been developed based on job sequencing, such as Shortest Processing Time rule (SPT), First Come First Serve rule (FCFS), Longest Processing Time rule (LPT), Page algorithm (Page, 1961), Palmer algorithm (Palmer, 1965), NEH heuristic (Nawaz et al., 1983), etc. However, prioritizing jobs still remains a challenge in flowshop scheduling, especially for today's mass customized production.

This paper focuses on the permutation flowshop scheduling problem (PFSP), which is commonly encountered in automotive manufacturing (Xu and Zhou, 2009), IC (Integrated Circuit) fabrication (Liu and Chang, 2000), photographic film production (Aghezzaf and Van Landeghem, 2002), pharmaceutical and agro-food industries (Boukef et al., 2007). Makespan minimization of PFSP, denoted by $Fm|prmu|C_{\max}$, has been proven NP-hard when the number of machines m is larger than three (Rinnooy Kan, 1976). NEH heuristic is deemed as the best constructive heuristic for PFSPs, and a number of NEH-based heuristics have been proposed (Dong et al., 2008; Kalczynski and Kamburowski, 2008; Li et al., 2004). There are two key steps in the NEH-framed algorithms: 1) order jobs; 2) insert jobs one by one. Step 1 is to define a suitable method to prioritize jobs based on their characteristics. Many different priority rules have been developed and tested. In the dominant NEH heuristic, the

* Corresponding author.
E-mail address: y.jin@qub.ac.uk (Y. Jin).

http://dx.doi.org/10.1016/j.ijpe.2017.06.026
Received 10 June 2016; Received in revised form 15 June 2017; Accepted 19 June 2017
Available online 21 June 2017

high priority is allocated to the job with a large sum of processing times. Framinan et al. (2003) confirmed the effectiveness of the priority rule of the NEH heuristic by testing 177 different initial sequences with makespan, idletime and flowtime criterion respectively. Li et al. (2004) proposed a new priority rule by accounting of the processing time deviation for the first time which resulted in better solutions. Dong et al. (2008) improved the priority rule further by using the average and standard deviation of processing times to order jobs and the approach has been proven effective (Fernandez-Viagas and Framinan, 2014; Ying and Lin, 2013). Kalczynski and Kamburowski proposed two priority rules, namely NEHKK1 (Kalczynski and Kamburowski, 2008) and NEHKK2 (Kalczynski and Kamburowski, 2009) based on Johnson's rule by assigning different weights to processing times.

The average of processing times is the only indicator in NEH priority rule while the standard deviation is included in NEH-D (Dong et al., 2008). In mathematics, the average and standard deviation stand for the first and second moments. But for hard problems, it is not enough to differentiate jobs with only two moments. Higher moment such as skewness should be used for further differentiation. The above mentioned moments have been successfully applied in image processing and retrieval (Kadir et al., 2011; Stricker and Orengo, 1995). The colour feature is fully characterized by adding the third moment, skewness, and the fourth moment, kurtosis. For example, if the colour model is RGB, the images can be differentiated effectively based on 12 moments with four moments for each channel. The moments have been successfully applied in image differentiation but they have not been studied in scheduling. Herein, the moments of processing time distribution in job ordering are studied. Some preliminary experiments have been conducted in our previous study (Liu et al., 2016), but the impact of each moment has not been investigated thoroughly. In this paper, a new priority rule including skewness is proposed and tested.

The second step is to insert jobs effectively. Based on the initial sequence, all possible inserting positions for unscheduled jobs will be tested against an objective, such as makespan or total flowtime. The location associated with the best objective value will be selected. This procedure will be repeated until all jobs are allocated. During the job insertion, ties often occur when jobs are inserted into different positions while sharing the same objective value. How to break ties has a direct effect on the solution quality and computation efficiency. In this paper, tie-breaking rules are classified into three categories according to the scope of the considered information and computational complexity: 1) job-based; 2) position-based; 3) schedule-based. The job-based tie-breaking rule only takes account of job processing time information. It is fast due to the small scale of job data. In NEHKK1 and NEHKK2, the inserting methods are two simple tie-breaking rules which belong to the first category. By assigning different job weights and determining the centre of gravity of job processing times, only the first or the last tie position is selected while the middle ones are not accounted. The effectiveness of NEHKK1 inserting method has been confirmed by Fernandez-Viagas and Framinan (2014).

In literature, most of tie-breaking rules (Liu et al., 2012; Nagano and Moccellin, 2002) fall into the second category given the tie position information. It is intuitive to break ties by using position information. The idle time on the bottleneck machine is considered in MNEH (Nagano and Moccellin, 2002) to choose tie position while in NEH-D, the position with balanced machine utilizations is selected. The position with the minimum completion times and balanced workloads at all machines is selected by Liu et al. (2012). In general, the position-based tie-breaking rule shows significant improvement compared to the NEH heuristic. It can be concluded that the tie-breaking rules from the first two categories do not increase computational complexity of the NEH heuristic but with slight increase of running time.

The third category is to break ties based on the partial schedule. In general, the solution quality could be further improved at the cost of computation time by considering a large scope of information. In NEHKK (Kalczynski and Kamburowski, 2007), the schedule with the least completion time between two tie positions (inclusive) is selected. Schedule-based information is taken into account in NEHKK, instead of job processing time or position-based information only. System idle time can also be used as the tie-breaking rule. The minimization of idle time (IT) between machines, as shown in Fig. 1, is used as the tie-breaking method by Ying and Lin (2013). A similar tie-breaking rule is developed to minimize IT (Companys et al., 2010), and if ties still exist, the tie-breaking method of NEHKK1 is adopted. However, IT minimization may not be consistent with makespan criterion (Liu et al., 2014). Front delay and IT minimization shows better performance. The idea is adopted in NEHFF heuristic (Fernandez-Viagas and Framinan, 2014) by rough calculation of system idle time. Its effectiveness has been validated on Taillard test bed (Taillard, 1993), but it is subject to the accuracy of the rough method. The complexity of these tie-breaking rules could be reduced to O(n^2m) if an approximation is applied.

In this paper, a new priority rule which enables subtle differentiation of processing time distributions is proposed by introducing the third central moment together with the average and standard deviation. A novel schedule-based tie-breaking rule is developed by minimizing front delay and partial IT before the new inserting position while maintaining the computational complexity of the NEH heuristic. Statistical test results illustrate better solution quality of the proposed priority rule, tie-breaking rule and heuristic respectively by comparing to existing rules and heuristics.

The remainder of this paper is organized as follows. In Section 2, the newly proposed heuristic is introduced. In Section 3, test cases and computational results are presented, demonstrating the effectiveness of the newly proposed heuristic. Final conclusions and future developments are presented in Section 4.

2. New heuristic

The PFSP is a NP-hard problem (Rinnooy Kan, 1976). This section introduces a new heuristic for the problem with makespan criterion, which is the most important and common measure used in industry (Framinan et al., 2004). The assumptions are described as follows.

1) All jobs are available at time of zero and start as soon as possible.
2) Processing time is known and deterministic.
3) Setup time is included in processing time.
4) Machines are continuously available but cannot process two or more jobs simultaneously.
5) Job pre-emption is not allowed.
6) Buffer capacity between machines is infinite.
7) Only permutation schedules are allowed.

The objective function can be expressed as

$$Min : F = C_{[n],m}. \quad (1)$$

The new priority rule, tie breaking rule and heuristic will be presented in the following sub-sections.

2.1. New priority rule: PR_{SKE}

Jobs in permutation flowshops are to be processed on a series of machines, and the processing times on each machine constitute a vector for each job. Analogous to the probability distribution, the moments of job processing times can be defined. In physics, moment stands for the combination of physical quantity and distance, such as torque. In

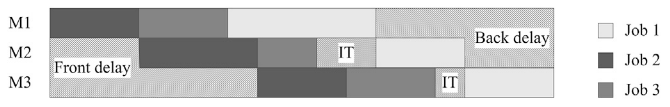


Fig. 1. Front delay, idle time (IT) and back delay (Spachis, 1978).

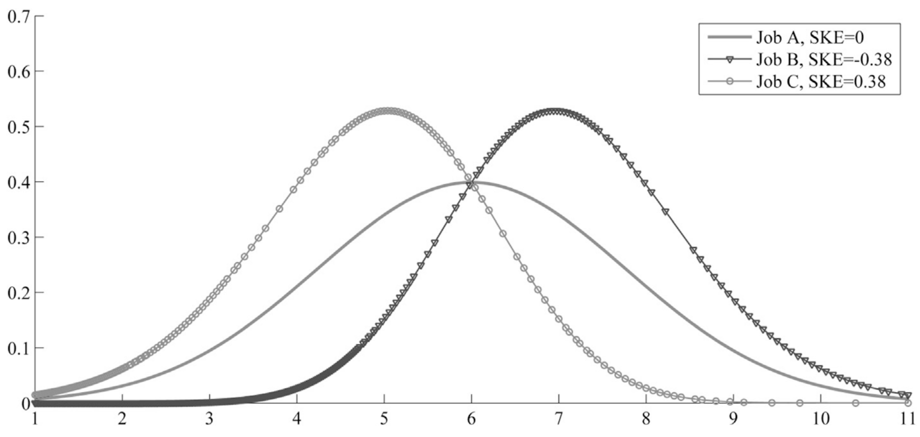


Fig. 2. Skewness in differentiating jobs with the same AVG_i and STD_i.

mathematics or statistics, moment is a quantitative measure to illustrate the shape of the distribution. Although moment has shown great potential in image differentiation, it has not been fully investigated in scheduling. In this study, the application of moments in job sorting is conducted and the effectiveness is validated in subsequent sections.

In the new priority rule PR_{SKE}, all jobs are ordered by the non-increasing sum of

$$AVG_i + STD_i + abs(SKE_i), \tag{2}$$

where AVG_i is the average processing times of job i, STD_i stands for the standard deviation of job i and $abs(SKE_i)$ represents the absolute value of the skewness of job i. The parameters are defined as follows.

$$AVG_i = \frac{1}{m}\sum_{k=1}^{m} t_{i,k}, \tag{3}$$

$$STD_i = \sqrt{\frac{1}{m-1}\sum_{k=1}^{m}(t_{i,k} - AVG_i)^2}, \tag{4}$$

$$SKE_i = \frac{\frac{1}{m}\sum_{k=1}^{m}(t_{i,k} - AVG_i)^3}{\left(\sqrt{\frac{1}{m}\sum_{k=1}^{m}(t_{i,k} - AVG_i)^2}\right)^3}. \tag{5}$$

Skewness SKE_i is a measure of asymmetry of a probability distribution. It can be positive or negative, as shown in Fig. 2. The positive skewness distribution has a long tail on the right side and most of data is concentrated on the left, e.g. job C. For a negative skewed distribution, its mass focuses on the right, represented by job B.

Skewness as the third moment is included in the new rule to differentiate jobs. As shown in Fig. 2, different jobs share the same average and standard deviation and they cannot be clearly differentiated by using NEH or NEH-D priority rule. For this reason, Gao et al. (2007) proposed 14 approaches to break ties for the NEH heuristic when sequencing jobs. Their results show that the NEH performance could be further improved if an appropriate method were applied. But it needs extra computation time to break ties in running priority rules. In the new rule PR_{SKE}, skewness combined with the average and standard deviation of processing times is used. The average, standard deviation and skewness stand for the first, second and third moments of processing times distribution respectively. These moments can characterize processing time distribution accurately and be used for job differentiation effectively. Ties can be significantly avoided due to the subtle characteristics by adding the third moment, skewness. If ties (with three terms added up) occur in an unlikely case, priority will be given to a job randomly.

For example, assume there are two jobs with processing time sets of [10, 3, 2, 6, 9] and [8, 10, 4, 7, 1]. The average and standard deviation of both jobs are the same (6 and 3.54, respectively), while their skewnesses are different (0 and −0.38), as shown in Fig. 2 represented by job A and job B. NEH and NEH-D could not cope with sorting ties, and random sequences of these two jobs may be obtained by different coding software. Due to the large impact of the priority rule, the solution quality may further vary after inserting all jobs. By using the new rule PR_{SKE}, the two jobs can be clearly differentiated by their skewnesses, 0 and −0.38.

A new hypothesis is proposed that the job with a larger absolute skewness should be given a high priority when two jobs have the same AVG and STD values. When a job has a skewed distribution, more processing times will deviate from the average than a job associated with a normal distribution. Therefore, job associated with the larger skewness should be given a high priority. As the positive and negative skewnesses of a distribution have the same influence herein, the absolute value of skewness is used.

All coefficients of each component in expression (2) are set as 1 in the new priority rule. The rationale is as follows. AVG_i is designed to be the main indicator when ordering jobs, followed by STD_i and SKE_i. Normally, the skewness value ranges from negative 3 to positive 3 and $SKE = 0$ if the distribution is symmetric, for example job A in Fig. 2. Due to its definition and range, the impact of skewness in the priority rule varies according to the scope and variation of sample data. In our case, all job processing times are in the range of [1,99] according to Taillard (1993) and VRF (Vallada et al., 2015) benchmarks, thus the AVG_i value should be around 50, followed by STD_i and SKE_i. Hence the coefficients of each component have been set as 1. To confirm this point, the weights ranging from [0.1, 0.2, …, 1] for three indicators are set and 1000 combinations are tested in total. As a result, the best three weight combinations are (0.8, 0.9, 0.6), (1, 1, 1) and (0.9, 1, 1) with ARPD values of 3.06, 3.06 and 3.07 respectively. The weight combination (1, 1, 1) is selected as it has the best performance working with the tie-breaking rule of NEH-D heuristic on Taillard benchmark.

2.2. New tie-breaking rule: TB_{LJP1}

To further improve the solution quality of the NEH heuristic, a novel tie-breaking rule named TB_{LJP1} is proposed. The new tie-breaking rule TB_{LJP1} is defined as: the sequence is chosen with the least sum of weighted job completion times and the least variation of gaps at the position where the job is newly inserted. Mathematically, the metric p is defined as

$$p = \sum_{k=1}^{m} w_k f_{x,k} + \alpha \sum_{k=1}^{m} \left| g_{x,k} - \overline{g} \right| \tag{6}$$

where w_k is the weight assigned to machine k, $f_{x,k}$ the job completion time at position x on machine k, x the position where the job is newly

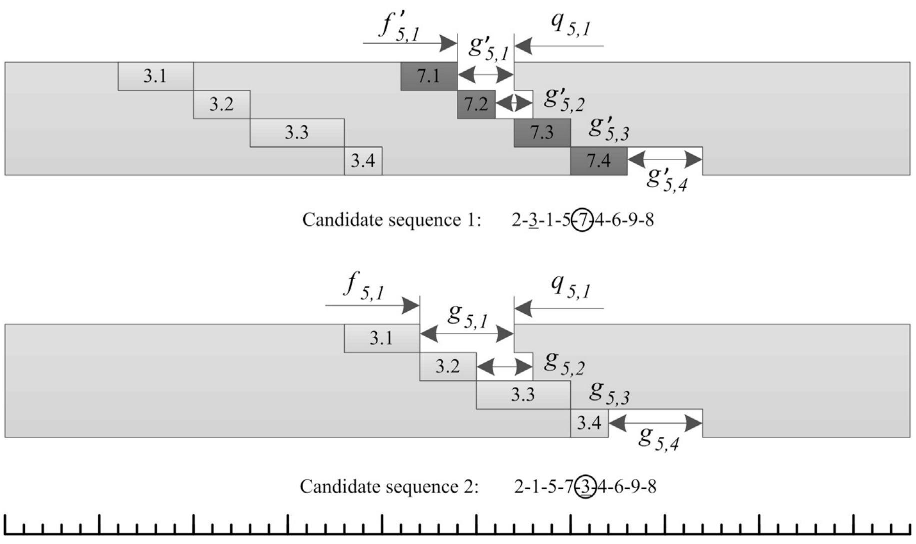


Fig. 3. Two candidate sequences with the same makespan (job 3 is inserted at the 2nd and 5th positions respectively, $x = 5$).

inserted, α the weight of the second indicator, $g_{x,k}$ the gap on machine k at position x where $g_{x,k} = C_{max} - f_{x,k} - q_{x,k}$, C_{max} the maximum completion time of the candidate sequence, $q_{x,k}$ the tail time on machine k defined by (Taillard, 1990), and $\bar{g}$ the average of $g_{x,k}$ on all machines.

The idea of the new tie-breaking rule is to minimize the front delay and IT, as shown in Fig. 1. However, the complexity will increase if the exact system idle time is calculated. A rough calculation method is proposed in NEHFF (Fernandez-Viagas and Framinan, 2014), but the effectiveness is subjected to its accuracy requirement. In this paper, an approach is introduced by calculating the front delay and partial IT before the new inserting position without increasing the computational complexity of the NEH heuristic O(n^2m). The proof is as follows.

The IT in both candidate sequences is divided into two parts by the job at the new inserting position. The front delay and partial IT before the new inserting position can be calculated by the job completion times $f_{x,k}$ on each machine. While the partial IT after the position can be minimized by a smooth gaps, as the job sequences after the new inserting position are kept the same in both candidate sequences. So the variation of gaps should be minimized. Herein, the mean absolute deviation (MAD) is taken as the variation measure in the new tie-breaking rule.

Note that the job completion times at the new inserting position have different impact on makespan. The job completion time on the 'critical' machine has a direct impact on makespan, whereas that on the non-critical machine has not. Therefore, the completion times are given different weights which are determined by the gaps between $f_{x,k}$ and $q_{x,k}$ (Taillard, 1990). A small gap on a machine indicates a large impact on makespan minimization. So the 'critical' machine with a gap of zero is assigned the highest weight. Based on the initial test, the weights of machines from low to high are defined as [0, 1, 2, …, m-1] as they are effective and robust on different benchmarks. Note that the gaps in the sequence where the job is newly inserted are used for determining the weights of machines, as it can directly reflect the job's impact on makespan.

By using the new tie-breaking rule, the computational complexity of the NEH heuristic does not increase, as explained with the example in Fig. 3. Job 3 is inserted into the 2nd and 5th positions respectively in the sequence of 2-1-5-7-4-6-9-8. Two candidate sequences are generated: 2-$\underline{3}$-1-5-7-4-6-8-9 and 2-1-5-7-$\underline{3}$-4-6-8-9. According to the new tie-breaking rule, the job completion times at the 5th position in both sequences are required. For sequence 2-1-5-7-$\underline{3}$-4-6-8-9, the completion times of job $\underline{3}$ can be calculated according to the Taillard acceleration, as well as the gaps. For sequence 2-$\underline{3}$-1-5-7-4-6-8-9 that job 3 is initially inserted, the completion times of job 7 at the 5th position can also be obtained without increasing the heuristic computational complexity. The steps of the calculation are given below according to the new tie-breaking rule.

1) Insert job 3 at the 2nd position, and calculate completion times $f'_{2,k}$ of job 3 in the sequence 2-3-1-5-7-4-6-8-9 according to the NEH acceleration;
2) Insert job 3 at the 3rd position, and calculate completion times of job 3 in the sequence 2-1-3-5-7-4-6-8-9 and the completion times $f'_{3,k}$ of job 1 in the sequence 2-$\underline{3}$-1-5-7-4-6-8-9;
3) Insert job 3 at the 4th position, and calculate completion times of job 3 in the sequence 2-1-5-3-7-4-6-8-9 and the completion times $f'_{4,k}$ of job 5 in the sequence 2-$\underline{3}$-1-5-7-4-6-8-9;
4) Insert job 3 at the 5th position, and calculate completion times $f_{5,k}$ of job 3 in the sequence 2-1-5-7-3-4-6-8-9 and the completion times $f'_{5,k}$ of job 7 in the sequence 2-$\underline{3}$-1-5-7-4-6-8-9.

The completion times of job 7 in the sequence 2-$\underline{3}$-1-5-7-4-6-8-9 are calculated step by step when different positions are tested for job 3. The gaps in both candidate sequences can be obtained as well. So, all parameters are available for the tie-breaking rule while maintaining the complexity of NEH heuristic. If the 5th inserting position is selected for job 3, $f_{5,k}$ will be saved, otherwise $f'_{5,k}$ is maintained.

According to the new tie-breaking rule, the job completion times at the new inserting position for sequence 2-$\underline{3}$-1-5-7-4-6-8-9 are 24, 26, 30 and 33, and for sequence 2-1-5-7-3-4-6-8-9, 22, 25, 30 and 32. The gaps in both sequences are 3, 2, 0, 4 and 5, 3, 0, 5. Note that the gaps 5, 3, 0, and 5 in the new candidate sequence are used to decide the weights of machines. So the weights of machine 1 to 4 are set as 1, 2, 3 and 0 as the weights are set in the range of [0, 1, …, m-1]. When the gaps are the same, the front machine will have the priority for weight setting. The sums of weighted completions times are 166 and 162 respectively, and the MADs are 1.25 and 1.75. So, the tie can be broken if the weight α is decided.

By combining the new priority rule and new tie-breaking rule, the new heuristic is developed, denoted by NEHLJP1.

3. Computational experiments

This section is to confirm the effectiveness of the third central moment in the new priority rule and test the performance of the new tie-breaking rule and the new heuristic. The recent NEH modifications including NEH-D (Dong et al., 2008), NEHKK1 (Kalczynski and

Table 1
ARPD values of different priority rules on Taillard benchmark.

Problem	NEH	PR_D	PR_{KK1} (with TB_{KK1})	PR_{KK2} (with TB_{KK2})	PR_{SKE}
20 × 5	3.30	2.70	2.81	**2.48**	2.71
20 × 10	4.60	4.08	4.43	4.17	**3.68**
20 × 20	3.73	3.82	3.34	3.57	**2.91**
50 × 5	0.73	0.89	0.67	**0.44**	0.88
50 × 10	5.07	4.90	5.46	5.38	**4.84**
50 × 20	6.65	**6.12**	6.25	6.22	6.42
100 × 5	0.53	0.41	0.41	**0.22**	0.54
100 × 10	2.21	2.16	**1.78**	2.28	2.24
100 × 20	5.34	5.65	5.23	5.32	**4.99**
200 × 10	1.26	1.27	1.32	**1.01**	1.24
200 × 20	4.41	4.57	4.17	4.25	**4.14**
500 × 20	2.07	2.12	**1.95**	2.01	2.12
AVG	3.32	3.22	3.15	3.11	**3.06**

The bold number is the best ARPD value for each problem.

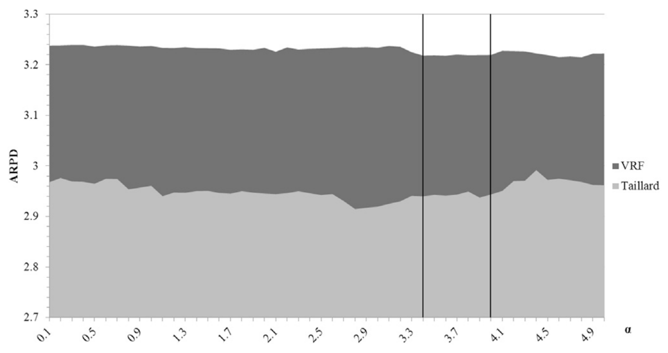


Fig. 4. ARPD values of TB_{LJP1} with $\alpha \in [0.1, 0.2, \ldots, 5]$.

Kamburowski, 2008), NEHKK2 (Kalczynski and Kamburowski, 2009) and tie-breaking method NEHFF (Fernandez-Viagas and Framinan, 2014) are taken as references. In this paper, only those NEH modifications with the computational complexity of $O(n^2m)$ are considered although many improvements or variants (Rad et al., 2009; Vasiljevic and Danilovic, 2015) have been developed.

Three sets of tests are conducted including comparison tests of the new priority rule, the new tie-breaking rule and the new heuristic respectively. The test benchmarks are taken from Taillard (1993) and VRF (Vallada et al., 2015) including 600 instances in total. All algorithms are coded in Matlab R2013b and run on a CPU i5-3210M computer with 4.00G memory.

In order to measure the solution quality of each algorithm, the relative percentage deviation (RPD), $RPD_p = \frac{HS_p - UB_p}{UB_p} \cdot 100\%$, is employed as the performance measure HS_p represents the value obtained by heuristics on problem instance p and UB_p the upper bound provided by Taillard (Taillard, n.d.) and VRF (Vallada et al., 2015) respectively. The details of each test are presented in the following subsections.

3.1. *Comparison tests of priority rules*

To check the impact of skewness in the priority rule and validate its effectiveness, the new priority rule is implemented in the NEH heuristic. For comparison, the original NEH heuristic, NEHKK1 priority rule PR_{KK1}, NEHD priority rule PR_D, and NEHKK2 priority rule PR_{KK2} are used as references. Note that the priority rules, PR_{KK1} and PR_{KK2}, are implemented together with their tie-breaking rules TB_{KK1} and TB_{KK2} respectively (Kalczynski and Kamburowski, 2009, 2008).

Table 1 shows the ARPD results of priority rules on Taillard test bed. The ARPD value of the new priority rule is 3.06, showing the best performance among all references. The test results of priority rules on VRF test bed are shown in Table 2. PR_{SKE} shows better performance than NEH and PR_D, and it has the same level performance as PR_{KK1} which is implemented with its tie-breaking rule. On both benchmarks, the new proposed priority rule shows better performance than PR_D by including the high order moment, i.e., skewness, for job differentiation.

3.2. *Comparison tests of tie-breaking rules*

The new tie-breaking rule is implemented within the NEH heuristic

Table 2
ARPD values of different priority rules on VRF benchmark.

Problem		NEH		PR_D		PR_{KK1} (with TB_{KK1})		PR_{KK2} (with TB_{KK2})		PR_{SKE}	
S	L	S	L	S	L	S	L	S	L	S	L
10 × 5	100 × 20	2.18	5.71	**1.51**	5.61	2.00	**5.37**	2.34	5.70	**1.51**	5.44
10 × 10	100 × 40	1.63	5.67	**1.46**	5.31	1.59	5.46	2.39	5.42	**1.46**	**5.25**
10 × 15	100 × 60	1.53	4.95	2.17	**4.51**	**1.48**	4.72	1.86	4.86	2.27	4.70
10 × 20	200 × 20	1.99	4.23	1.52	4.04	1.54	4.18	**1.50**	**4.00**	1.62	4.13
20 × 5	200 × 40	1.51	4.71	2.76	4.66	**1.13**	**4.49**	2.43	4.76	2.82	4.56
20 × 10	200 × 60	4.82	4.55	4.93	4.35	5.07	**4.30**	**4.77**	4.31	4.80	4.45
20 × 15	300 × 20	4.33	3.00	3.93	3.03	4.23	2.85	4.10	**2.82**	**3.89**	3.00
20 × 20	300 × 40	4.12	4.08	**3.50**	3.90	3.91	**3.80**	4.22	3.99	3.51	3.97
30 × 5	300 × 60	1.43	3.93	1.64	3.91	**0.92**	**3.84**	1.07	4.24	1.54	3.85
30 × 10	400 × 20	5.26	2.58	5.46	2.46	5.09	2.39	5.22	**2.21**	**4.73**	2.47
30 × 15	400 × 40	5.83	3.66	5.44	**3.51**	6.02	3.70	5.53	3.60	**5.24**	3.57
30 × 20	400 × 60	**5.41**	3.56	5.49	3.47	5.44	3.42	5.61	3.49	5.57	**3.36**
40 × 5	500 × 20	1.09	2.27	0.79	2.23	0.61	1.84	**0.45**	**1.73**	0.93	2.00
40 × 10	500 × 40	4.97	3.20	4.52	3.11	4.78	3.09	5.01	3.02	**4.13**	**3.00**
40 × 15	500 × 60	6.05	3.12	5.87	3.20	6.14	**3.09**	6.33	3.19	5.74	3.10
40 × 20	600 × 20	**5.14**	1.57	5.29	1.64	5.35	**1.50**	5.37	1.52	5.38	1.61
50 × 5	600 × 40	0.55	3.13	0.82	2.98	0.45	3.00	**0.37**	**2.88**	0.80	2.97
50 × 10	600 × 60	4.58	2.93	4.45	2.94	4.30	**2.86**	**3.48**	2.94	4.26	2.91
50 × 15	700 × 20	6.52	1.40	6.90	1.23	6.36	1.33	**6.19**	**1.11**	6.21	1.30
50 × 20	700 × 40	5.96	2.77	6.00	2.60	6.38	2.68	6.10	**2.45**	**5.70**	2.65
60 × 5	700 × 60	0.89	2.75	0.48	2.71	0.77	**2.68**	**0.18**	2.76	0.55	2.69
60 × 10	800 × 20	3.96	1.23	**3.94**	1.15	4.06	1.13	4.02	**0.95**	4.39	1.13
60 × 15	800 × 40	5.79	2.43	5.91	2.52	**5.69**	2.44	5.96	**2.32**	5.95	2.40
60 × 20	800 × 60	6.45	2.71	6.42	2.67	**6.08**	2.58	6.68	**2.56**	6.57	2.57
AVG(S)	AVG(L)	3.83	3.34	3.80	3.24	**3.72**	**3.20**	3.80	**3.20**	3.73	3.21
AVG		3.59		3.52		**3.46**		3.50		3.47	

The bold number is the best ARPD value for each problem.

Table 3
ARPD values of different tie-breaking rules on Taillard benchmark.

Problem	NEH	TB_D	TB_{KK1}	TB_{KK2}	TB_{FF}	TB_{LJP1}
20 × 5	3.30	2.48	2.73	2.65	**2.29**	2.36
20 × 10	4.60	**4.13**	4.31	4.31	4.15	4.73
20 × 20	3.73	3.70	3.41	3.41	**3.30**	3.34
50 × 5	0.73	0.73	0.59	0.66	0.92	**0.56**
50 × 10	5.07	4.80	4.87	4.83	5.15	**4.69**
50 × 20	6.65	6.24	6.41	6.37	6.21	**6.11**
100 × 5	0.53	0.49	0.40	0.42	0.42	**0.36**
100 × 10	2.21	1.96	1.77	1.86	2.17	**1.62**
100 × 20	5.34	**5.01**	5.28	5.30	5.02	5.09
200 × 10	1.26	1.01	1.17	1.12	0.97	**0.93**
200 × 20	4.41	3.88	4.17	4.24	4.07	**3.78**
500 × 20	2.07	**1.70**	2.02	2.00	1.76	1.71
AVG	3.32	3.01	3.09	3.10	3.04	**2.94**

The bold number is the best ARPD value for each problem.

on both Taillard and VRF benchmarks. The weight α is selected from the range [0.1, 0.2, 0.3, …, 5.0] and the ARPD values of TB_{LJP1} with different α is shown in Fig. 4. The performance of the new tie-breaking rule varies on both benchmarks due to random data but it remains good and robust in the range of [3.4, 3.5, …, 4.0]. To select a precise value for α, the new tie-breaking is run with the new priority rule PR_{SKE} rule on Taillard benchmark, and the best combination is chosen at $\alpha = 3.4$.

Similar to the comparison test of priority rules, different tie-breaking rules are applied in the NEH heuristic when ties occur. Existing popular tie-breaking rules including TB_D (Dong et al., 2008), TB_{KK1} (Kalczynski and Kamburowski, 2008), TB_{KK2} (Kalczynski and Kamburowski, 2009) and TB_{FF} (Fernandez-Viagas and Framinan, 2014) are taken as references.

As shown in Table 3, all tie-breaking rules improve NEH performance on Taillard test bed. TB_D, TB_{KK1}, TB_{KK2} and TB_{FF} achieve 3.01, 3.09, 3.10 and 3.04 respectively while the newly proposed tie-breaking rule TB_{LJP1} outperforms all existing ones with an ARPD value of 2.94, achieving the best on 7/12 problems among all references. On VRF benchmark, the solution quality of NEH heuristic is significantly improved by using the new tie-breaking rule TB_{LJP1} with an ARPD value of 3.22. On small instances, TB_{LJP1} achieves 3.48, and on large instances 2.96, both better than other references as shown in Table 4. By comparing the results in Tables 1–4, it can be seen that TB_{KK1} and TB_{KK2} have the same or even

Table 4
ARPD values of different tie-breaking rules on VRF benchmark.

Problem		NEH		TB_D		TB_{KK1}		TB_{KK2}		TB_{FF}		TB_{LJP1}	
S	L	S	L	S	L	S	L	S	L	S	L	S	L
10 × 5	100 × 20	2.18	5.71	2.31	5.50	2.21	5.80	2.23	5.76	2.54	5.35	**1.92**	**5.10**
10 × 10	100 × 40	1.63	5.67	1.46	5.25	1.74	5.52	1.74	5.43	1.71	**5.10**	**1.14**	5.15
10 × 15	100 × 60	1.53	4.95	1.60	**4.73**	1.54	4.92	1.44	4.86	1.31	4.91	**1.30**	5.16
10 × 20	200 × 20	1.99	4.23	1.75	**3.82**	**1.54**	4.00	**1.54**	4.09	1.70	3.87	1.78	3.84
20 × 5	200 × 40	1.51	4.71	1.24	**4.41**	1.19	4.55	2.09	4.51	**1.14**	4.62	1.16	4.64
20 × 10	200 × 60	4.82	4.55	**4.04**	4.26	4.71	4.40	4.71	4.40	4.79	4.45	4.42	**4.14**
20 × 15	300 × 20	4.33	3.00	3.91	**2.47**	4.02	2.80	4.02	2.79	**3.83**	2.60	3.86	2.49
20 × 20	300 × 40	4.12	4.08	3.85	3.70	3.69	3.89	**3.64**	3.87	4.05	3.77	3.76	**3.56**
30 × 5	300 × 60	1.43	3.93	1.03	3.73	1.09	3.73	1.24	3.76	0.91	3.77	**0.80**	**3.69**
30 × 10	400 × 20	5.26	2.58	5.61	2.11	**4.87**	2.33	5.18	2.40	5.35	1.98	5.00	**1.97**
30 × 15	400 × 40	5.83	3.66	**5.32**	3.20	5.75	3.50	5.76	3.45	5.56	3.33	5.45	**3.10**
30 × 20	400 × 60	5.41	3.56	5.44	**3.29**	5.43	3.31	5.43	3.33	5.09	3.40	**4.91**	3.39
40 × 5	500 × 20	1.09	2.27	0.91	1.63	**0.63**	1.95	0.81	1.91	0.77	1.66	0.75	**1.52**
40 × 10	500 × 40	4.97	3.20	**4.08**	**2.70**	5.07	3.05	5.02	3.01	4.15	2.78	4.20	2.80
40 × 15	500 × 60	6.05	3.12	5.98	**2.83**	6.14	3.13	6.05	3.09	**5.22**	3.05	5.58	2.93
40 × 20	600 × 20	**5.14**	1.57	5.18	**1.24**	5.68	1.43	5.61	1.56	5.15	**1.24**	5.32	1.27
50 × 5	600 × 40	0.55	3.13	0.47	**2.45**	0.47	2.86	0.32	2.92	**0.31**	2.65	0.39	2.51
50 × 10	600 × 60	4.58	2.93	3.97	2.66	4.43	2.82	4.22	2.81	3.77	2.75	**3.68**	**2.57**
50 × 15	700 × 20	6.52	1.40	6.40	1.08	6.46	1.27	6.44	1.24	6.29	1.07	**5.95**	**1.02**
50 × 20	700 × 40	5.96	2.77	**5.88**	**2.30**	5.98	2.76	5.99	2.65	6.26	**2.30**	5.94	2.32
60 × 5	700 × 60	0.89	2.75	0.78	**2.41**	0.70	2.76	0.67	2.81	0.62	2.54	**0.61**	2.46
60 × 10	800 × 20	3.96	1.23	3.51	**0.88**	3.89	1.19	3.96	1.19	**3.38**	0.98	3.84	0.96
60 × 15	800 × 40	5.79	2.43	**5.13**	**1.98**	5.76	2.45	5.78	2.50	5.66	2.15	5.47	2.02
60 × 20	800 × 60	6.45	2.71	**6.11**	2.37	6.25	2.64	6.42	2.67	6.32	2.38	6.28	**2.36**
AVG(S)	AVG(L)	3.83	3.34	3.58	**2.96**	3.72	3.21	3.76	3.21	3.58	3.03	**3.48**	**2.96**
AVG		3.59		3.27		3.46		3.49		3.30		**3.22**	

The bold number is the best ARPD value for each problem.

Table 5
ARPD values of each combination on Taillard benchmark.

Problem	PR_D					PR_{SKE}				
	TB_{KK1}	TB_D	TB_{KK2}	TB_{FF}	TB_{LJP1}	TB_{KK1}	TB_D	TB_{KK2}	TB_{FF}	TB_{LJP1}
20 × 5	2.65	2.81	2.65	2.56	2.19	2.60	2.23	2.42	2.36	2.16
20 × 10	3.81	3.75	3.81	3.54	3.79	3.81	3.77	3.81	3.56	3.68
20 × 20	3.60	3.64	3.60	3.33	3.53	3.27	3.00	3.27	3.16	3.06
50 × 5	0.82	0.73	0.80	0.75	0.63	0.80	0.91	0.98	0.80	0.64
50 × 10	5.25	4.66	5.25	4.90	4.64	4.75	5.01	4.64	5.17	4.25
50 × 20	5.89	5.81	5.92	5.81	5.78	6.42	5.95	6.40	6.49	6.15
100 × 5	0.45	0.40	0.43	0.41	0.35	0.45	0.53	0.49	0.47	0.36
100 × 10	2.01	1.70	1.99	1.69	1.46	2.23	1.98	1.82	1.93	1.72
100 × 20	5.50	4.98	5.40	5.15	5.00	5.03	4.93	5.02	5.08	4.81
200 × 10	1.14	0.96	1.25	0.96	0.99	1.15	1.08	1.12	1.03	0.89
200 × 20	4.27	3.75	4.26	3.92	3.86	4.37	3.96	4.16	3.83	3.65
500 × 20	1.96	1.66	2.06	1.75	1.72	1.96	1.65	1.87	1.72	1.62
AVG	3.11	2.91	3.12	2.90	2.83	3.07	2.92	3.00	2.97	2.75

Table 6
ARPD values of each combination on VRF benchmark.

Problem		PR_D										PR_{SKE}									
		TB$_{KK1}$		**TB**$_D$		**TB**$_{KK2}$		**TB**$_{FF}$		**TB**$_{LJP1}$		**TB**$_{KK1}$		**TB**$_D$		**TB**$_{KK2}$		**TB**$_{FF}$		**TB**$_{LJP1}$	
S	L	S	L	S	L	S	L	S	L	S	L	S	L	S	L	S	L	S	L	S	L
10 × 5	100 × 20	1.72	5.44	1.26	5.25	1.61	5.60	1.50	5.41	1.22	5.20	1.72	5.35	1.35	5.23	1.61	5.20	1.81	5.27	1.44	5.22
10 × 10	100 × 40	1.53	5.30	1.88	5.27	1.53	5.30	1.63	5.17	1.54	5.49	1.53	5.44	1.88	5.42	1.53	5.35	1.63	5.04	1.54	5.22
10 × 15	100 × 60	2.27	4.59	2.13	4.51	2.09	4.59	2.32	4.71	2.13	4.71	2.19	4.66	2.22	4.70	2.19	4.66	2.34	4.61	2.25	4.63
10 × 20	200 × 20	1.79	3.80	1.81	3.66	1.79	3.80	1.51	3.84	1.72	3.53	1.90	4.01	1.91	3.71	1.90	4.00	1.61	3.79	1.82	3.76
20 × 5	200 × 40	2.59	4.53	2.96	4.34	2.88	4.43	3.07	4.33	2.61	4.39	2.51	4.58	2.90	4.31	2.78	4.57	2.86	4.15	2.89	4.24
20 × 10	200 × 60	4.91	4.18	4.96	4.17	4.91	4.18	5.09	4.17	4.77	4.17	4.94	4.25	4.85	4.15	4.85	4.25	5.02	4.15	4.72	4.21
20 × 15	300 × 20	3.95	2.62	3.82	2.38	3.95	2.65	3.62	2.61	3.90	2.45	4.03	2.76	3.79	2.34	4.03	2.73	3.61	2.37	3.81	2.56
20 × 20	300 × 40	3.39	4.12	3.32	3.60	3.38	4.06	3.27	3.70	3.43	3.45	3.44	3.74	3.46	3.47	3.34	3.70	3.46	3.56	3.22	3.41
30 × 5	300 × 60	1.07	3.88	1.15	3.84	1.14	3.87	0.96	3.78	1.04	3.80	1.06	3.85	1.41	3.62	1.12	3.86	0.88	3.76	1.03	3.54
30 × 10	400 × 20	4.74	2.21	4.76	1.89	4.74	2.24	4.65	1.92	5.07	1.78	4.81	2.22	4.82	1.78	4.87	2.08	4.46	1.88	4.43	1.89
30 × 15	400 × 40	5.40	3.46	4.53	3.08	5.40	3.51	5.00	3.31	4.94	3.15	5.46	3.55	4.63	2.94	5.46	3.57	4.37	3.20	5.11	3.13
30 × 20	400 × 60	5.54	3.29	5.14	3.16	5.54	3.33	5.34	3.25	5.27	3.24	5.98	3.21	5.32	3.22	5.98	3.22	5.20	3.09	5.38	3.12
40 × 5	500 × 20	0.86	1.82	0.74	1.61	0.81	1.92	0.91	1.51	0.83	1.54	0.88	1.95	0.76	1.64	0.74	1.94	0.60	1.62	0.53	1.54
40 × 10	500 × 40	4.08	2.89	4.04	2.56	4.08	2.96	3.85	2.72	4.13	2.59	4.11	2.95	3.65	2.51	4.11	3.00	4.28	2.80	4.05	2.73
40 × 15	500 × 60	5.48	3.14	5.30	3.01	5.34	3.15	5.58	3.09	5.19	2.95	5.38	3.07	5.16	2.93	5.69	3.07	5.10	3.00	5.34	2.96
40 × 20	600 × 20	5.51	1.49	5.14	1.27	5.51	1.55	5.12	1.21	5.32	1.13	5.33	1.55	5.50	1.22	5.33	1.46	5.75	1.24	5.12	1.24
50 × 5	600 × 40	0.54	2.77	0.39	2.48	0.47	2.78	0.49	2.48	0.50	2.48	0.54	2.77	0.56	2.44	0.60	2.78	0.49	2.54	0.47	2.43
50 × 10	600 × 60	3.77	2.91	3.78	2.53	3.87	2.95	3.56	2.60	3.30	2.62	3.97	2.93	3.67	2.52	4.02	2.93	3.97	2.55	3.03	2.61
50 × 15	700 × 20	6.32	1.20	5.53	0.94	6.20	1.17	6.03	1.08	6.07	0.97	6.75	1.19	6.14	1.02	6.86	1.27	5.96	1.00	5.79	0.97
50 × 20	700 × 40	6.00	2.47	5.89	2.23	6.00	2.50	5.68	2.17	5.31	2.17	5.62	2.54	5.73	2.23	5.62	2.61	6.08	2.26	5.23	2.29
60 × 5	700 × 60	0.83	2.71	0.43	2.35	0.60	2.63	0.45	2.32	0.30	2.34	0.76	2.54	0.58	2.25	0.75	2.59	0.25	2.51	0.27	2.31
60 × 10	800 × 20	3.72	1.11	3.32	0.88	3.60	1.10	3.71	0.87	3.54	0.94	3.31	1.14	3.82	0.81	3.34	1.16	3.30	0.93	3.27	0.91
60 × 15	800 × 40	5.95	2.23	5.82	2.02	5.86	2.31	5.87	2.04	5.88	1.97	6.12	2.34	5.72	1.95	6.02	2.30	5.60	2.05	5.51	1.92
60 × 20	800 × 60	6.48	2.59	5.65	2.36	6.40	2.65	5.69	2.36	5.97	2.42	6.44	2.55	5.82	2.33	6.44	2.52	5.97	2.40	6.02	2.30
AVG(S)	AVG(L)	3.68	3.11	3.49	2.89	3.65	3.13	3.54	2.94	3.50	2.90	3.70	3.13	3.57	2.86	3.72	3.12	3.53	2.91	3.43	2.88
AVG		3.40		3.19		3.39		3.24		3.20		3.42		3.22		3.42		3.22		3.15	

Table 7
ARPD values of each heuristic on Taillard benchmark.

Problem	NEH	NEH-D	NEHKK1	NEHKK2	NEHLJP1
20 × 5	3.30	2.81	2.81	2.48	**2.16**
20 × 10	4.60	3.75	4.43	4.17	**3.68**
20 × 20	3.73	3.64	3.34	3.57	**3.06**
50 × 5	0.73	0.73	0.67	**0.44**	0.64
50 × 10	5.07	4.66	5.46	5.38	**4.25**
50 × 20	6.65	**5.81**	6.25	6.22	6.15
100 × 5	0.53	0.40	0.41	**0.22**	0.36
100 × 10	2.21	**1.70**	1.78	2.28	1.72
100 × 20	5.34	4.98	5.23	5.32	**4.81**
200 × 10	1.26	0.96	1.32	1.01	**0.89**
200 × 20	4.41	3.75	4.17	4.25	**3.65**
500 × 20	2.07	1.66	1.95	2.01	**1.62**
AVG	3.32	2.91	3.15	3.11	**2.75**

The bold number is the best ARPD value for each problem.

better performance than NEHKK1 and NEHKK2 which means that the effectiveness of NEHKK1 and NEHKK2 are resulted from their tie-breaking rules rather than the priority rules.

In order to further investigate the performance of each priority rule and tie-breaking rule, tests including all combinations are conducted on Taillard and VRF benchmarks.

Table 5 shows the results of all combinations on Taillard benchmark. The effectiveness of different tie-breaking rules can be illustrated by the results in each main column. For example, in the main column of PR_D, the tie-breaking rules TB_D, TB_{KK1}, TB_{KK2}, TB_{FF} and the new rule TB_{LJP1} achieve 3.11, 2.91, 3.12, 2.90 and 2.83 respectively. Similarly, the effectiveness of each priority rule can be seen from the comparison of each sub-column. By using the new priority rule of PR_{SKE}, the performance of TB_{KK1} and TB_{KK2} is notably improved comparing with their original priority rules, PR_{KK1} and PR_{KK2}. With the new tie-breaking rule TB_{LJP1}, the priority rule PR_D generates a better ARPD value than with its original tie-breaking rule.

From Table 6, similar conclusions can be drawn on VRF benchmark. By using the new priority rule, TB_{KK1} and TB_{KK2} are significantly enhanced and the performance of TB_D is maintained. With PR_{SKE}, TB_{FF} generates the best solutions than with other priority rules. When running all priority rules together with the new tie-breaking rule, significant improvements can be seen, demonstrating the effectiveness of TB_{LJP1}.

Table 9
Paired samples *t*-test results for tie-breaking rules on the combined benchmark.

Pairs	Mean	SEM	IC-lower	IC-upper	T	Sig.
NEH - TB_{LJP1}	0.00372	0.00030	0.00314	0.00430	12.584	0.000
TB_D - TB_{LJP1}	0.00055	0.00027	0.00002	0.00109	2.044	0.041
TB_{KK1} - TB_{LJP1}	0.00227	0.00030	0.00168	0.00287	7.526	0.000
TB_{KK2} - TB_{LJP1}	0.00246	0.00032	0.00183	0.00308	7.769	0.000
TB_{FF} - TB_{LJP1}	0.00088	0.00027	0.00034	0.00141	3.204	0.001

3.3. Comparison tests of heuristics

As shown in Table 7, the new heuristic NEHLJP1 outperforms the existing NEH improvements on 8/12 problems of Taillard benchmark with an ARPD value of 2.75. It achieves the second best among all heuristics for the rest of problems. While on VRF test bed as shown in Table 8, NEHLJP1 also dominates existing ones with an overall ARPD value of 3.15, 3.43 on small instances and 2.88 on large instances respectively.

3.4. Significant difference

In order to check if the differences of ARPDs among tie-breaking rules or heuristics are statistically significant, a paired-samples *t*-test is carried out. Note that the paired *t*-test is conducted on a large number of instances by combining the Taillard and VRF benchmarks to decrease the influence of sample size, as the *t*-test results is affected by the size of benchmark (Fernandez-Viagas and Framinan, 2014; Kalczynski and Kamburowski, 2008).

The paired *t*-test results in terms of tie-breaking rules are listed in Table 9. It can be seen from the results that when the confidence level α is defined as 0.05, there are significant differences in terms of ARPD between the newly proposed tie-breaking rule TB_{LJP1} and other reference tie-breaking rules. Table 10 shows the significance results of heuristics.

Table 8
ARPD values of each heuristic on VRF benchmark.

Problem		NEH		NEH-D		NEHKK1		NEHKK2		NEHLJP1	
S	L	S	L	S	L	S	L	S	L	S	L
10 × 5	100 × 20	2.18	5.71	**1.26**	5.25	2.00	5.37	2.34	5.70	1.44	**5.22**
10 × 10	100 × 40	1.63	5.67	1.88	5.27	1.59	5.46	2.39	5.42	**1.54**	**5.22**
10 × 15	100 × 60	1.53	4.95	2.13	**4.51**	**1.48**	4.72	1.86	4.86	2.25	4.63
10 × 20	200 × 20	1.99	4.23	1.81	**3.66**	1.54	4.18	**1.50**	4.00	1.82	3.76
20 × 5	200 × 40	1.51	4.71	2.96	4.34	**1.13**	4.49	2.43	4.76	2.89	**4.24**
20 × 10	200 × 60	4.82	4.55	4.96	**4.17**	5.07	4.30	4.77	4.31	**4.72**	4.21
20 × 15	300 × 20	4.33	3.00	3.82	**2.38**	4.23	2.85	4.10	2.82	**3.81**	2.56
20 × 20	300 × 40	4.12	4.08	3.32	3.60	3.91	3.80	4.22	3.99	**3.22**	**3.41**
30 × 5	300 × 60	1.43	3.93	1.15	3.84	**0.92**	3.84	1.07	4.24	1.03	**3.54**
30 × 10	400 × 20	5.26	2.58	4.76	**1.89**	5.09	2.39	5.22	2.21	**4.43**	**1.89**
30 × 15	400 × 40	5.83	3.66	**4.53**	**3.08**	6.02	3.70	5.53	3.60	5.11	3.13
30 × 20	400 × 60	5.41	3.56	**5.14**	3.16	5.44	3.42	5.61	3.49	5.38	**3.12**
40 × 5	500 × 20	1.09	2.27	0.74	1.61	0.61	1.84	**0.45**	1.73	0.53	**1.54**
40 × 10	500 × 40	4.97	3.20	**4.04**	**2.56**	4.78	3.09	5.01	3.02	4.05	2.73
40 × 15	500 × 60	6.05	3.12	**5.30**	3.01	6.14	3.09	6.33	3.19	5.34	**2.96**
40 × 20	600 × 20	5.14	1.57	5.14	1.27	5.35	1.50	5.37	1.52	**5.12**	**1.24**
50 × 5	600 × 40	0.55	3.13	0.39	2.48	0.45	3.00	**0.37**	2.88	0.47	**2.43**
50 × 10	600 × 60	4.58	2.93	3.78	**2.53**	4.30	2.86	3.48	2.94	**3.03**	2.61
50 × 15	700 × 20	6.52	1.40	**5.53**	**0.94**	6.36	1.33	6.19	1.11	5.79	0.97
50 × 20	700 × 40	5.96	2.77	5.89	**2.23**	6.38	2.68	6.10	2.45	**5.23**	2.29
60 × 5	700 × 60	0.89	2.75	0.43	2.35	0.77	2.68	0.18	2.76	**0.27**	**2.31**
60 × 10	800 × 20	3.96	1.23	3.32	**0.88**	4.06	1.13	4.02	0.95	**3.27**	0.91
60 × 15	800 × 40	5.79	2.43	5.82	2.02	5.69	2.44	5.96	2.32	**5.51**	**1.92**
60 × 20	800 × 60	6.45	2.71	**5.65**	2.36	6.08	2.58	6.68	2.56	6.02	**2.30**
AVG(S)	AVG(L)	3.83	3.34	3.49	2.89	3.72	3.20	3.80	3.20	**3.43**	**2.88**
AVG		3.59		3.19		3.46		3.50		**3.15**	

The bold number is the best ARPD value for each problem.

Table 10
Paired samples *t*-test results.

Benchmark	Pair	Mean	SEM	IC-lower	IC-upper	T	Sig.
Combined benchmark	NEH - NEHLJP1	0.00461	0.00039	0.00384	0.00538	11.761	0.000
	NEH-D - NEHLJP1	0.00060	0.00030	0.00002	0.00118	2.023	0.044
	NEHKK1 - NEHLJP1	0.00326	0.00037	0.00252	0.00399	8.711	0.000
	NEHKK2 - NEHLJP1	0.00350	0.00040	0.00271	0.00428	8.741	0.000
Taillard	NEH - NEHLJP1	0.00574	0.00082	0.00411	0.00737	6.971	0.000
	NEH-D - NEHLJP1	0.00154	0.00078	-0.00001	0.00310	1.973	0.051
	NEHKK1 - NEHLJP1	0.00400	0.00082	0.00237	0.00563	4.863	0.000
	NEHKK2 - NEHLJP1	0.00361	0.00086	0.00192	0.00531	4.215	0.000
VRF	NEH - NEHLJP1	0.00432	0.00044	0.00345	0.00520	9.744	0.000
	NEH-D - NEHLJP1	0.00036	0.00031	-0.00025	0.00098	1.158	0.247
	NEHKK1 - NEHLJP1	0.00307	0.00042	0.00225	0.00390	7.318	0.000
	NEHKK2 - NEHLJP1	0.00347	0.00045	0.00258	0.00436	7.669	0.000

Table 11
Summary of heuristics in terms of ARPD, ACPU and ARPT.

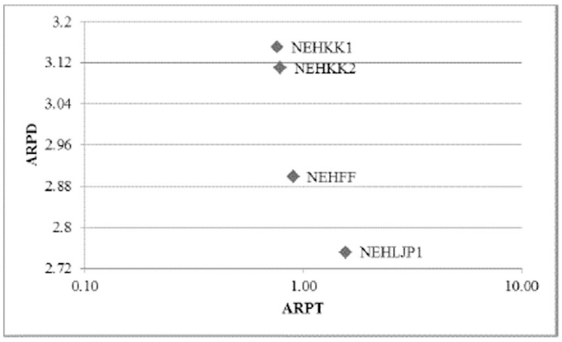
Algorithm	Taillard			VRF		
	ARPD	ACPU	ARPT	ARPD	ACPU	ARPT
NEHKK1	3.15	2.05	0.76	3.46	19.30	0.85
NEHKK2	3.11	2.11	0.78	3.50	19.31	0.85
NEHFF	2.90	2.29	0.90	3.24	19.70	0.96
NEHLJP1	2.75	3.15	1.56	3.15	22.94	1.34

All p values are less than 0.05, demonstrating the significant differences present between the new heuristic and existing ones. The t-test is also conducted on both benchmarks separately, showing that the differences between NEHLJP1 and NEH, NEHKK1, NEHKK2 are significant. The p-values between NEHLJP1 and NEH-D are 0.051 and 0.247 on Taillard and VRF benchmarks respectively which mean the differences between them on two benchmarks are not significant.

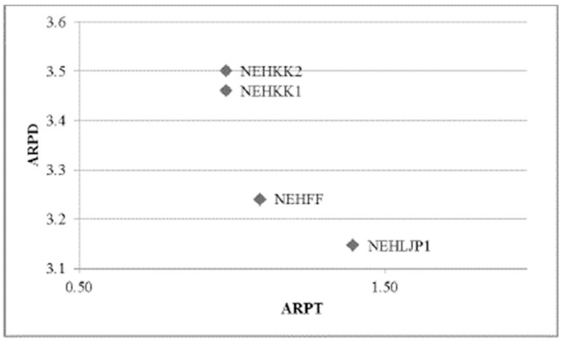
3.5. Computation efficiency

In order to check the computation efficiency of the new heuristic, the ARPD, the average CPU time (ACPU) and the average relative percentage time (ARPT) are used simultaneously (Fernandez-viagas et al., 2017). The ACPU and ARPT of heuristic p are defined as $ACPU_p = \frac{\sum_{\forall j} CPU_{j,p}}{J}$, $ARPT_p = \sum_{\forall j} \frac{CPU_{j,p} - ACT_j}{ACT_j} \Big/ J + 1$, where $CPU_{j,p}$ is the computation time on instance j by heuristic p, J the total number of instances, $ACT_j = \frac{\sum_{\forall p} CPU_{j,p}}{P}$, P the total number of heuristics considered. ACPU and ARPT are two performance measures of computation efficiency, though the solution quality is also to be considered as a key performance factor which is represented by ARPD.

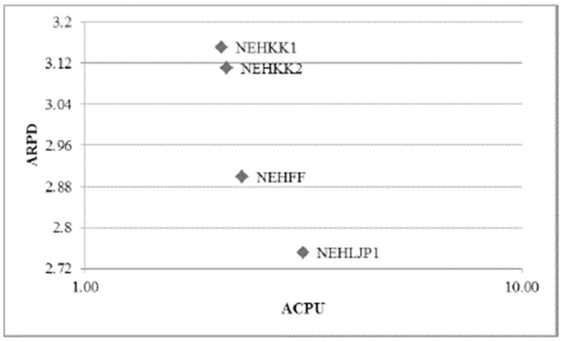
Each heuristic is run for ten times independently and the maximum value is removed. As stated in (Fernandez-viagas et al., 2017), NEHKK2 and the heuristic NEHFF implemented with PR_D and TB_{FF} are deemed as efficient, and they are taken as references in this paper. NEHKK1 is



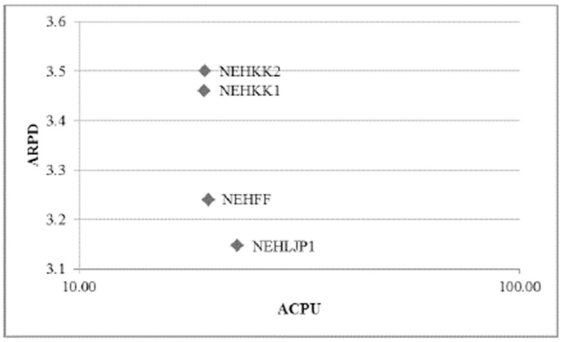
(a) ARPD vs. ARPT of heuristics in logarithmic scale on Taillard benchmark



(b) ARPD vs. ARPT of heuristics in logarithmic scale on VRF benchmark



(c) ARPD vs. ACPU of heuristics in logarithmic scale on Taillard benchmark



(d) ARPD vs. ACPU of heuristics in logarithmic scale on VRF benchmark

Fig. 5. Computation efficiency of heuristics.

included as well as it shows better performance than NEHKK2 on VRF benchmark within a shorter computation duration.

Table 11 shows the results on both benchmarks. NEHLJP1 is associated with the lowest ARPD values on both benchmarks at the cost of computation time. To compare the efficiency of heuristics, the figures of ARPD vs. ARPT and ARPD vs. ACPU are also given, as shown in Fig. 5 on each benchmark, which clearly shows in all cases that NEHLJP1 has the best solution quality but worst computation efficiency in comparing to NEHFF, NEHKK2 and NEHKK1. It is worth noting that the average computation times of the four algorithms are always in the same magnitude on each benchmark.

4. Conclusions

The third moment of distribution of job processing times is introduced for job differentiation. It is the first time to study the effects by including different moments for characterizing processing time distributions in scheduling. By using the first three moments, average, standard deviation and skewness, jobs can be clearly differentiated and ordered. The test results on different benchmarks showed the effectiveness of the newly introduced priority rule. The fourth moment, kurtosis, is also studied, and results show there is no improvement comparing to PR_{SKE}. So in the priority rule for the NEH heuristic, the moments higher than the third order are not included.

A novel tie-breaking rule for the NEH heuristic is proposed by minimizing front delay and partial IT before the tie position without increasing the computational complexity. Different weights are assigned to machines based on the schedule relaxations and the minimum variation of gaps is pursued. Test results show that it outperforms existing rules significantly.

In conclusion, a new heuristic named NEHLJP1 is proposed for the permutation flowshop scheduling problem with makespan criterion by combining the new priority rule and tie-breaking rule. The effectiveness of the new heuristic has been validated on both Taillard and VRF test beds.

Acknowledgement

The authors would like to thank the editor and anonymous referees for their valuable, constructive, and very helpful comments and suggestions. We also thank the authors of NEHKK1, NEHKK2, NEH-D and TB_{FF}, for providing data and algorithms. The Ph.D. scholarship supported by Queen's University Belfast, United Kingdom and China Scholarship Council (201306050018) is acknowledged.

References

Aghezzaf, E., Van Landeghem, H., 2002. An integrated model for inventory and production planning in a two-stage hybrid production system. Int. J. Prod. Res. 40, 4323–4339.

Boukef, H., Benrejeb, M., Borne, P., 2007. A proposed genetic algorithm coding for flow-shop scheduling problems. Int. J. Comput. Commun. Control 2, 229–240.

Companys, R., Ribas, I., Mateo, M., 2010. Improvement tools for NEH based heuristics on permutation and blocking flow shop scheduling problems. In: IFIP Advances in Information and Communication Technology, pp. 33–40.

Dong, X., Huang, H., Chen, P., 2008. An improved NEH-based heuristic for the permutation flowshop problem. Comput. Oper. Res. 35, 3962–3968.

Fernandez-Viagas, V., Framinan, J.M., 2014. On insertion tie-breaking rules in heuristics for the permutation flowshop scheduling problem. Comput. Oper. Res. 45, 60–67.

Fernandez-viagas, V., Ruiz, R., Framinan, J.M., 2017. A new vision of approximate methods for the permutation flowshop to minimise makespan: State-of-the-art and computational evaluation. Eur. J. Oper. Res. 257, 707–721.

Framinan, J.M., Gupta, J.N.D., Leisten, R., 2004. A review and classification of heuristics for permutation flow-shop scheduling with makespan objective. J. Oper. Res. Soc. 55, 1243–1255.

Framinan, J.M., Leisten, R., Rajendran, C., 2003. Different initial sequences for the heuristic of Nawaz, Enscore and Ham to minimize makespan, idletime or flowtime in the static permutation flowshop sequencing problem. Int. J. Prod. Res. 41, 121–148.

Gao, S., Liu, Y., Zhang, W., Zhang, J., 2007. Some supplementation for NEH heuristic in solving flow shop problem. In: Proceedings of the American Control Conference. New York, pp. 3348–3352.

Kadir, A., Nugroho, L.E., Susanto, A., Santosa, P.I., 2011. Leaf classification using shape, color, and texture features. Int. J. Eng. Trends Technol. 2, 225–230.

Kalczynski, P.J., Kamburowski, J., 2007. On the NEH heuristic for minimizing the makespan in permutation flow shops. Omega 35, 53–60.

Kalczynski, P.J., Kamburowski, J., 2008. An improved NEH heuristic to minimize makespan in permutation flow shops. Comput. Oper. Res. 35, 3001–3008.

Kalczynski, P.J., Kamburowski, J., 2009. An empirical analysis of the optimality rate of flow shop heuristics. Eur. J. Oper. Res. 198, 93–101.

Li, X., Wang, Y., Wu, C., 2004. Heuristic algorithms for large flowshop scheduling problems. In: Fifth World Congress on Intelligent Control and Automation. Hangzhou, China, pp. 2999–3003.

Liu, C.Y., Chang, S.C., 2000. Scheduling flexible flow shops with sequence-dependent setup effects. IEEE Trans. Robot. Automation 16, 408–419.

Liu, G., Song, S., Wu, C., 2012. Two techniques to improve the NEH algorithm for flow-shop scheduling problems. In: Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence, pp. 41–48.

Liu, W., Jin, Y., Price, M., 2014. A new heuristic to minimize system idle time for flowshop scheduling. In: Poster Presented at the 3rd Annual EPSRC Manufacturing the Future Conference. Glassogw, September 23–24.

Liu, W., Jin, Y., Price, M., 2016. A new Nawaz–Enscore–Ham-based heuristic for permutation flow-shop problems with bicriteria of makespan and machine idle time. Eng. Optim. 48, 1808–1822.

Nagano, M.S., Moccellin, J.V., 2002. A high quality solution constructive heuristic for flow shop sequencing. J. Oper. Res. Soc. 53, 1374–1379.

Nawaz, M., Enscore Jr., E.E., Ham, I., 1983. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. Omega 11, 91–95.

Page, E.S., 1961. An approach to the scheduling of jobs on machines. J. R. Stat. Soc. 23, 484–492.

Palmer, D.S., 1965. Sequencing jobs through a multi-stage process in the minimum total time-a quick method of obtaining a near optimum. Oper. Res. Q. 16, 101–107.

Rad, S.F., Ruiz, R., Boroojerdian, N., 2009. New high performing heuristics for minimizing makespan in permutation flowshops. Omega 37, 331–345.

Rinnooy Kan, A.H.G., 1976. Machine Scheduling Problems: Classification, Complexity and Computations. Martinus Nijhoff, The Hague, The Netherlands.

Spachis, A.S., 1978. Job-shop Scheduling with Approximate Methods. PhD diss.. Imperial College London (University of London).

Stricker, M., Orengo, M., 1995. Similarity of color images. In: Proc. SPIE Storage and Retrieval for Image and Video Databases III 2420, pp. 381–392.

Taillard, E., 1990. Some efficient heuristic methods for the flow shop sequencing problem. Eur. J. Oper. Res. 47, 65–74.

Taillard, E., 1993. Benchmarks for basic scheduling problems. Eur. J. Oper. Res. 64, 278–285.

Taillard, E., n.d. Summary of best known lower and upper bounds of Taillard's instances [WWW Document]. URL http://mistic.heig-vd.ch/taillard/problemes.dir/ordonnancement.dir/flowshop.dir/best_lb_up.txt (accessed 5.22.15).

Vallada, E., Ruiz, R., Framinan, J.M., 2015. New hard benchmark for flowshop scheduling problems minimising makespan. Eur. J. Oper. Res. 240, 666–677.

Vasiljevic, D., Danilovic, M., 2015. Handling ties in heuristics for the permutation flow shop scheduling problem. J. Manuf. Syst. 35, 1–9.

Xu, J., Zhou, X., 2009. A class of multi-objective expected value decision-making model with birandom coefficients and its application to flow shop scheduling problem. Inf. Sci. 179, 2997–3017.

Ying, K., Lin, S., 2013. A high-performing constructive heuristic for minimizing makespan in permutation flowshops. J. Ind. Prod. Eng. 30, 355–362.