Effective ensembles of heuristics for scheduling flexible job shop problem with new job insertion

Research · September 2015

DOI: 10.13140/RG.2.1.4157.7445

1 author:

Kaizhou Gao
Nanyang Technological University
36 PUBLICATIONS **173** CITATIONS

SEE PROFILE

Computers & Industrial Engineering 90 (2015) 107–117

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Effective ensembles of heuristics for scheduling flexible job shop problem with new job insertion

Kai Zhou Gao [a,c,*], Ponnuthurai Nagaratnam Suganthan [a], Mehmet Fatih Tasgetiren [b], Quan Ke Pan [c], Qiang Qiang Sun [d]

[a] School of Electrical and Electronic Engineering, Nanyang Technological University, 639798 Singapore, Singapore
[b] Department of Industrial Engineering, Yasar University, Bomova, Izmir, Turkey
[c] School of Computer, Liaocheng University, Liaocheng 252059, PR China
[d] Department of Information Engineering, Binzhou University, Binzhou 256603, PR China

ARTICLE INFO

Article history:
Received 9 October 2014
Received in revised form 14 August 2015
Accepted 7 September 2015
Available online 10 September 2015

Keywords:
Flexible job shop scheduling
Ensemble
Heuristic
New job insertion
Multiple objectives

ABSTRACT

This study investigates the flexible job shop scheduling problem (FJSP) with new job insertion. FJSP with new job insertion includes two phases: initializing schedules and rescheduling after each new job insertion. Initializing schedules is the standard FJSP problem while rescheduling is an FJSP with different job start time and different machine start time. The time to do rescheduling is the same as the time of new job insertion. Four ensembles of heuristics are proposed for scheduling FJSP with new job insertion. The objectives are to minimize maximum completion time (makespan), to minimize the average of earliness and tardiness (E/T), to minimize maximum machine workload (Mworkload) and total machine workload (Tworkload). Extensive computational experiments are carried out on eight real instances from remanufacturing enterprise. The results and comparisons show the effectiveness of the proposed heuristics for solving FJSP with new job insertion.

1. Introduction

Flexible job shop scheduling problem (FJSP) is an extension of classical job shop scheduling problem (JSP) (Jain & Meeran, 1998). FJSP includes two sub-problems, machine assignment and operation sequence. Machine assignment is to select a processing machine from candidate machines for each operation. Operation sequence is to schedule all operations on all machines to obtain feasible and satisfactory solutions. Hence, FJSP is very complicated and have been proven to be an NP-hard problem (Garey, Johnson, & Sethi, 1976).

Brucker and Schlie (1990) first study FJSP problem and proposed a polynomial algorithm for FJSP problem with two jobs. In recent years, many research works solve FJSP problem using heuristics and meta-heuristics. Heuristics include machine assignment component and operation sequence component. For machine assignment component, Pezzella, Morganti, and Ciaschetti (2008) proposed operation minimum processing time heuristic and global minimum processing time heuristic. Li, Pan, and Gao (2011) and Li, Pan, Suganthan, and Chua (2011) developed operation minimum processing time heuristic to generate initial solutions. Vilcot and Billaut (2011) proposed two-step greedy rule. Gao, Suganthan, and Pan (2014) mixed operation minimum processing time rule and earliest available machine rule to construct a new heuristic. For operation sequence component, Brandimarte (1993) proposed most work remaining and shortest processing time heuristics. Pezzella et al. (2008) proposed most number of operations remaining heuristic. The advantage of simple heuristics is their ability to find a feasible solution in a very short time. Simple heuristics cannot ensure obtaining the optimal solution or approximately optimal solutions. For meta-heuristics, tabu (TS) (Li, Pan, & Liang, 2010), genetic algorithm (GA) (Gao, Sun, & Gen, 2008), partial swarm optimization (PSO), artificial bee colony (ABC) (Li et al., 2010; Wang, Zhou, Xu, Wang, & Liu, 2012), estimation of distribution algorithm (EDA) (Wang, Wang, Xu, & Liu, 2013) and harmony search algorithm (HS) (Gao et al., 2014; Yuan & Xu, 2013) were employed for solving FJSP with an objective or multiple objectives. Generally, meta-heuristics can obtain better quality solutions than simple heuristics. However, meta-heuristics need longer time than simple heuristics, especially for large problem.

FJSP exists in many industry fields, such as mechanical manufacturing, remanufacturing, semiconductor manufacturing, and automobile assembly process. There are many in these industry

* Corresponding author at: School of Electrical and Electronic Engineering, Nanyang Technological University, 639798 Singapore, Singapore.

http://dx.doi.org/10.1016/j.cie.2015.09.005
0360-8352/

fields. However, most existing literatures do not consider practical constraints and uncertainty related issues in real industrial environments. Few researchers focused on FJSP problem considering real-life processing constraints. Mousakhani (2013) considered sequence-dependent setup time in FJSP with total tardiness. A mathematical model was developed to formulate FJSP with sequence-dependent setup time and an iteration based metaheuristic was proposed. Wang, Wang, and Liu (2013) and Wang, Zhou, Xu, and Liu (2013) studied FJSP with fuzzy processing time using ABC and estimation of distribution algorithm (EDA). The influence of parameter setting was considered in both ABC and EDA. A left-shift scheme was employed for improving the scheduling solution in decoding stage. In addition, crossover and variable neighborhood search (VNS) were employed for improving the performance of ABC. Xiong, Xing, and Chen (2013) researched robust scheduling multi-objective FJSP with random machine breakdowns. Two surrogate measures for robustness were developed. One was for machine breakdown and another was for the location of float times and machine breakdown at the same time. Al-Hinai and ElMekkawy (2011) researched robust and stable scheduling for FJSP with random machine breakdowns using a two-stage hybrid genetic algorithm. The first stage considered the standard FJSP while the second stage was for machine breakdown in the decoding space. In addition, Calleja and Pastor (2014) and Wang, Yin, and Qin (2013) studied FJSP with considerations for transfer of batches and machine disruption.

This study researches on FJSP problem with new job insertion. This problem is modeled from remanufacturing environments. New job insertion is one of seven features of remanufacturing. In remanufacturing environment, the account of returned products and the return time are factors that cannot be controlled by remanufacturers. There may be job(s) coming and be inserted into the current solution when the solution is being executed. In this condition, new job(s) and non-started operations of existing jobs will have to be rescheduled. The start time of different jobs may be different and the start time of different machines may also be different. Hence, the initial scheduling phase is the standard FJSP problem while the problem in rescheduling phase is an FJSP with different start times for different machines depending on their completion times of on-going operations. Rescheduling time should be very short to make sure continue processing in shop floor. We proposed several ensembles of heuristics for solving FJSP with new job insertion constraint. Experiment results show that these ensembles of heuristics can obtain better quality solutions than simple heuristics and do not need long time as metaheuristics. The objectives are to minimize maximum completion time (makespan), to minimize the average of earliness and tardiness (E/T), and to minimize the maximum machine workload (Mworkload) and total workload (Tworkload). The discussions and comparisons show the performance of ensembles of heuristics for solving FJSP with new job insertion.

The remainder of this paper is organized as follows. Section 2 describes the FJSP with new job insertion. In Section 3, the heuristics and proposed ensembles of heuristics are presented in detail. Experimental design, comparisons and discussions are presented in Section 4. We conclude this paper in Section 5.

2. FJSP with new job insertion

2.1. Flexible job shop problem

In FJSP, each job includes a sequence of operations. An operation can be processed on one set of candidate machines. One operation must be processed only on one machine with no interruption, while one machine can process only one operation at a time. The following notations and assumptions are used for the formulation of FJSP:

(1) Let $J = \{J_i\},\ 1 \leqslant i \leqslant n$, indexed i, be a set of n jobs to be scheduled. q_i denotes the total number of operations of job J_i.
(2) Let $M = \{M_k\},\ 1 \leqslant k \leqslant m$, indexed k, be a set of m machines.
(3) Each job J_i consists of a predetermined sequence of operations. Let $O_{i,h}$ be operation h of J_i.
(4) Each operation $O_{i,h}$ can be processed without interruption on one of a set of candidate machines $M(O_{i,h})$.

Let $P_{i,h,k}$ be the processing time of $O_{i,h}$ on machine M_k.

(5) Decision variables

$$x_{i,h,k} = \begin{cases} 1, & \text{if machine } k \text{ is selected for operation } O_{i,h} \\ 0, & \text{otherwise} \end{cases}$$

$c_{i,h}$ denotes the completion time of operation $O_{i,h}$
c_i denotes the completion time of job J_i
(6) The objectives are to minimize Makespan, E/T, Mworkload and Tworkload.

Makespan, denoted by C_M, can be calculated as follows:

$$\text{Min } C_M = \max_{1 \leqslant i \leqslant n}\{C_i\} \tag{1}$$

where c_i is the completion time of job i.

Average of earliness and tardiness, denoted by E/T, is the earliness or tardiness of job J_i compared to the due date of job J_i.

$$\text{Min } E/T = \frac{\sum_{i=1}^{n}|c_i - d_i|}{n} \tag{2}$$

where c_i is the completion time of job J_i and d_i is the due date of job i.

Maximum workload, denoted by W_M, can be calculated by:

$$\text{Min } W_M = \max_{1 \leqslant j \leqslant m}\{w_j\} \tag{3}$$

Total workload, denoted by W_T, can be calculated by:

$$\text{Min } W_T = \sum_{j=1}^{m} w_j \tag{4}$$

where w_j is the workload of machine j.

2.2. New job insertion

New job insertion constraint is modeled from remanufacturing environment (Daniel & Guide, 2000; Ferguson, 2009). Rescheduling operator is necessary when a new job comes and is inserted into the scheduled solution that is being executed in shop floor. For existing job, the start time is the insertion time or the completion time of current processing operation. For each machine, start time is the insertion time or the completion time of the on-going operation on this machine. Hence, the problem in rescheduling phase is FJSP with different job start times and different machine start times.

To explain FJSP problem with a new job insertion, an example is shown in Fig. 1. Fig. 1(a) shows a Gantt chart for 3-jobs and 3-machines FJSP problem. The number operations in Job1, Job2, and Job3 are 3, 2 and 2, respectively. The makespan value in this scheduling solution is 10. The completion times of M1, M2, and M3 are 8, 10 and 8. Fig. 1(b) shows the new job, Job4, comes and will be inserted into the executing schedule at Time 3. Job4 has three operations. Fig. 1(c) shows the result with no rescheduling

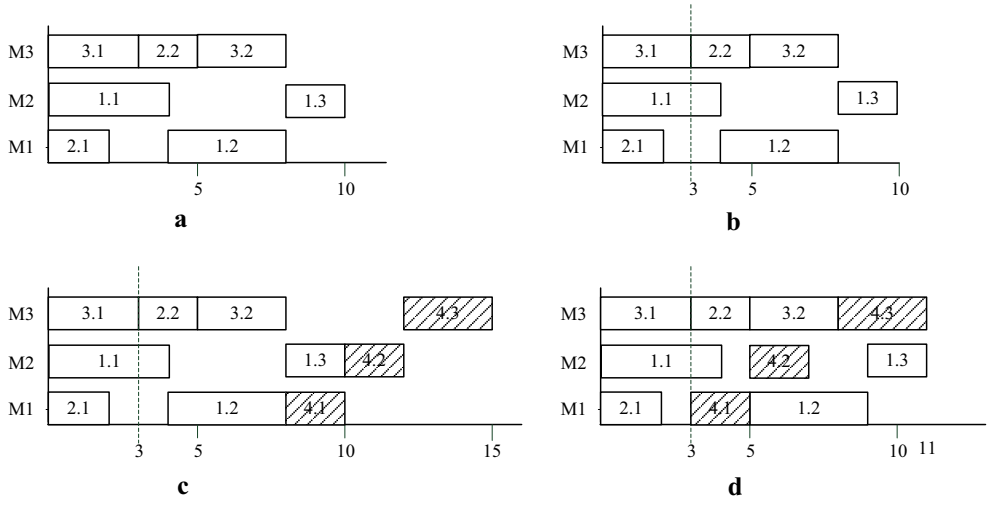

Fig. 1. An example of new job insertion.

while inserting Job4 directly. Existing scheduling scheme is retained and the Job4 is scheduled when the last operation on each machine is completed. The start times of three machines (M1, M2 and M3) for Job4 are 8, 10 and 8. The processing start time of Job4 operations on three machines (M1, M2 and M3) are 8, 10, and 12. The completion time of Job4 is 15. Fig. 1(d) shows rescheduling solution. Both new Job4 and all non-started operations of existing three jobs are rescheduled after Job4 inserted at Time 3. The available start time of M1 and M3 is Time 3, while the available start time of Job2, Job3 and Job4 is also Time 3. M2 is processing the first operation of Job1 when Job4 is inserted at Time 3. M2 and Job1 are available when the first operation of Job1 is completed. Therefore, the available start time of M2 and Job1 is Time 4. The rescheduling solution is shown in Fig. 1(d). The makespan value is 11. The start time of Operation 1.2 and Operation 1.3 are delayed than those in Fig. 1(b). For Job4, the processing times of all three operations are advanced than those in Fig. 1(c).

3. Heuristics and ensembles of heuristics

3.1. Encoding and decoding

An FJSP solution includes two parts, machine assignment (MA) and operation sequence (OS). MA is a machine sequence for each operation while OS is operation sequence for order operations on corresponding machines. Fig. 2 shows an example of MA and OS in an FJSP solution. Fig. 2(a) illustrates an MA vector while Fig. 2 (b) shows the corresponding OS. In MA, each element is the machine selected for the corresponding operation. For example, the first value "4" in MA vector means the fourth machine is selected for operation O1,1. In OS, each element means one operation of corresponding job. For example, the first value "3" in OS vector represents the first operation of job 3. The second value "2" means the first operation of job 2. The third value "1" is the first operation of job 1. The fourth value "2" means the second operation of job 2.

The decoding is to convert the solution to a feasible scheduling solution or a scheduling Gantt chart. The coded solution in Fig. 2 can be decoded to a Gantt chart as shown in Fig. 3.

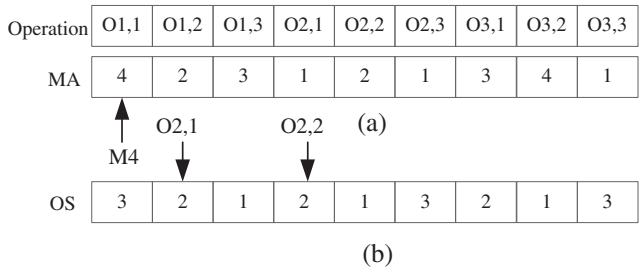

Fig. 2. An example of coded solution.

3.2. Simple heuristics

There are many simple heuristics in existing literatures for both machine assignment and operation sequencing.

For machine assignment, there are five heuristics as follows:

1. Operation minimum processing time (MA1) (Pezzella et al., 2008): For each operation O_{ij}, the machine M with minimum processing time will be selected from candidate machine set and be placed at the corresponding position in MA. In this rule, all operations will be assigned to the machine with minimum processing time. For a single operation, the processing time is minimal. However, there may be many operations queuing on the same machine. The machine workload and makespan objectives may be affected.
2. Local minimum processing time heuristic (MA2) (Pezzella et al., 2008): This heuristic developed operation minimum processing time rule. This heuristic adds the processing time for current operation O_{ij} to machine workload. When the same machine is selected for another operation O_{pq}, the selection criterion is the sum of machine workload and processing time, denoted

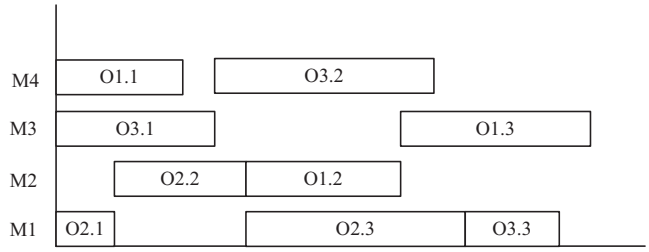

Fig. 3. Gantt chart after decoding.

by T_{sum}. The machine with minimum T_{sum} value will be selected for operation O_{pq}. This rule considers single operation processing time and machine workload.

3. Global minimum processing time heuristic (MA3) (Pezzella et al., 2008): This heuristic selects the minimum processing time from all-machine processing time for all operations. Similar to the second heuristic, the processing time is also added to machine workload. This heuristic starts from finding the global minimum processing time and considers machine workload. The disadvantage is lack of diversity.
4. Minimum completion time (MA4) (Gao et al., 2014): This heuristic was proposed in our previous work. In this heuristic, we compare completion times of two machines, one with minimum processing time and another with earliest start time. The one with smaller completion time will be selected for current operation. This rule considers the completion time of each operation and conducive to the optimization of the makespan.
5. Two-step greedy heuristic (MS5) (Vilcot & Billaut, 2011): The operations are sorted in ascending order based on the number of selectable machines. Ascending order of processing times is used to break ties when there is equal number of selectable machines. The machines are sorted using their workload in non-decreasing order. The operation is taken from the operations list and the first machine that belongs to the machine list is assigned to the operation. The workload of this machine is updated and the machine sorting is also updated. The process iterates until all operations have been assigned to one machine. The two-step greedy rule is proposed for minimization of the maximum tardiness. The performance of this heuristic for makespan and workload objectives is unsatisfactory.

For operation sequence, there are two heuristics as follows:

1. Most work remaining rule (OS1) (Brandimarte, 1993): This heuristic firstly orders the jobs in a descending order based on the remaining work. The job with the most remaining work will be selected first and put into OS. The iteration is repeated until all operations of all jobs have been sequenced. This method based on the remaining work. The remaining work calculated from the processing machine and processing time. Hence, machine assignment must be decided before this heuristic.
2. Most operations remaining rule (OS2) (Brandimarte, 1993): This heuristic selects jobs based on the number of remaining operations. The job with the most remaining un-sequenced operations will be selected first. This method relies on the total number of remaining operations rather than machine assignment.

3.3. Ensembles of heuristics

In this section, we propose an ensemble of four heuristics based on the advantages and disadvantages of the simple heuristics in Section 3.2.

EH1:

Step1: Generate all-machine processing time table for all operations.
Step2: Randomly order all operations of all jobs regardless of the operation priority in the same job.
Step3: Obtain MA using MA2 heuristic with the operation order in Step2.
Step4: Calculate the total workload of each job based on MA.
Step5: Generate OS using OS1 heuristic with the workload in Step4.

EH1 improves the randomness of machine assignments. Because OS1 relies on MA, this ensemble of heuristic also improves the randomness of operation sequence.

EH2:

Step1: Generate all-machine processing time for all operations in a table.
Step2: Randomly order all operations of all jobs regardless of the operation priority in the same job.
Step3: Obtain MA using MA2 heuristic with the operation order in Step2.
Step4: Calculate the total number of operations for each job.
Step5: Generate OS using OS2 heuristic with the total number of operations in Step4.

EH2 also improves the randomness of machine assignments. Because OS2 relies on the total number of operations of each job, this ensemble of heuristic takes the total number of operations as one considered factor.

EH3:

Step1: Generate MA by randomly selecting processing machine from the candidate machine set.
Step2: Count the total workload of each job based on MA.
Step3: Obtain OS using OS1 heuristic with the workload in Step2. Random selection is used to break ties.
Step4: For each operation in OS, select one machine using MA4 and obtain MA′.
Step5: Compare the completion times between MA and MA′ with the same OS.
Step6: Select machine assignment with smaller completion time as the final MA.

EH3 considers workload and makespan objectives in operation sequence and machine assignment, respectively.

EH4:

Step1: Count the total number of operations for each job.
Step2: Generate OS using OS2 heuristic with the operation number in Step1 and random selection is used to break ties.
Step3: For each operation in OS, select one machine using MA4.

EH4 consider both the total number of operations of all jobs and makespan objective.

To explain the four ensembles of heuristics further, we present an FJSP example with 3-jobs, 4-machines, and 9-operations solved by EH3. The processing times of different machines for different operations are shown in Table 1.

Step1: MA is generated randomly, MA = {4,2,3,1,2,1,3,4,1}. In MA, nine operations are assigned to four different machines.

Table 1
An example of 3-jobs, 4-machines FJSP problem.

Job	Operation	M1	M2	M3	M4
Job1	O11	2	3	–	4
	O12	4	5	3	6
	O13	–	5	6	3
Job2	O21	2	–	6	5
	O22	6	4	2	–
	O23	7	4	5	5
Job3	O31	6	3	5	4
	O32	5	6	4	7
	O33	3	6	5	4

Step2: Machine processing time for corresponding operation can be obtained from Table 1, T = {4,5,6,2,4,7,5,7,3}. The total workload of each job can also be computed, W = {4 + 5 + 6,2 + 4 + 7,5 + 7 + 3} = {15,13,15}.

Step3: Job1 and Job3 have the same workload of 15. Job3 is selected randomly and the first operation of Job3 is placed in OS, OS = {3}. The remaining workload of Job3 reduces from 15 to 10, W = {4 + 5 + 6,2 + 4 + 7,7 + 3}. Job1 has the maximum remaining workload and the first operation of Job1 is placed in OS, OS = {3,1}. The remaining workload of Job1 reduces from 15 to 11, W = {5 + 6,2 + 4 + 7,7 + 3}. Job2 has the maximum remaining workload and the first operation of Job2 is placed in OS, OS = {3,1,2}. The remaining workload of Job2 reduces from 13 to 11, W = {5 + 6,4 + 7,7 + 3}. Job1 and Job2 have the same remaining workload of 11. Job2 is selected randomly and the second operation of Job2 is placed in OS, OS = {3,1,2,2}. The remaining workload of Job2 reduces from 11 to 7, W = {5 + 6,7,7 + 3}. Job1 has the maximum remaining workload and the second operation of Job1 is placed in OS, OS = {3,1,2,2,1}. The remaining workload of Job1 reduces from 11 to 6, W = {6,7,7 + 3}. Then, the second operation of Job3 is placed in OS, OS = {3,1,2,2,1,3}. The remaining workload of Job3 is 3, W = {6,7,3}. The three final operations of three jobs are placed in OS based on the processing time. The operation sequence OS is obtained, OS = {3,1,2,2,1,3,2,1,3}.

Step4: The first operation in OS is O_{31}. The start times of all machines are ST = {0,0,0,0}. Machine M2 has the minimum processing time, 3, and M2 is selected for O_{31}, and the start time of M2 changes from 0 to 3, ST = {0,3,0,0}. The second operation in OS is O_{11}. Machine M1 has the minimum processing time, 2, and M1 is selected for O_{11}. The start time of M1 increase from 0 to 2, ST = {2,3,0,0}. The third operation in OS is O_{21}. Machine M1 has the minimum processing time, 2. Machines M3 and M4 have the minimum start time, 0. The completion time of O_{21} on machine M1, M3 and M4 are 4, 6, 5. Hence, M1 is selected for O_{21} and the start time of M1 increase from 2 to 4, ST = {4,3,0,0}. The fourth operation in OS is O_{22}. Machine M3 has the minimum processing time, 2 and earliest start time, 0. M3 is selected for O_{22} and the start time of M3 increase from 0 to 2, ST = {4,3,2,0}. The fifth operation in OS is O_{12}. Machine M3 has the minimum processing time, 3 and machine M4 has the earliest start time, 0. The completion time on M3 and M4 are 5 and 6. Hence, M3 is selected for O_{12} and the start time of M3 increases from 2 to 5, ST = {4,3,5,0}. The sixth operation in OS is O_{32}. Machine M3 has the minimum processing time, 4 and machine M4 has the earliest start time, 0. The completion time on M3 and M4 are 9 and 7. Hence, M4 is selected for O_{32} and the start time of M4 increase from 0 to 7, ST = {4,3,5,7}. Based on the same heuristic rule, the last operations are assigned to machines M2, M4, M1 and the final machine assignment is MA′ = {2,1,1,3,3,4,2,4,1}.

Step5: The two machine assignments are MA = {4,2,3,1,2,1,3,4,1}, MA′ = {2,1,1,3,3,4,2,4,1}, and operation sequence is OS = {3,1,2,2,1,3,2,1,3}. The completion time of MA and MA′ are 17 and 13, respectively.

Step6: Machine assignment MA′ = {2, 1,1,3, 3,4, 2, 4,1} is selected as the final MA.

4. Computational results and comparisons

4.1. Experimental setup

To test the performance of heuristics and proposed ensembles of heuristics, extensive experimental evaluation and comparisons are provided using instances from a remanufacturing enterprise. These instances include eight instances with the size ranging from 5-jobs, 4-machines and 23-operations to 20-jobs, 15-machines and 355-operations. 35 new jobs will be inserted into existing scheduling sequence. Each insertion has a different insertion time, job number and total number of operations. The detail information is shown in Table 2. The first two columns are instance number and new job insertion order. The third column is the insertion time of corresponding insertion order. The last three columns are the corresponding job number, machine number and the total number of operations. The first row of each instance shows the job number, machine number and operation number at initializing scheduling.

All simple heuristics and ensembles of heuristics are coded in C ++ and implemented on an Inter® Core™2 Duo CPU P8600 @ 2.40 GHz PC with 4 GB RAM. Each instance is carried out 30 replications by each heuristic.

4.2. Initializing scheduling

In this section, we compare 11 simple and ensembles of heuristics. These heuristics include three simple heuristics for machine assignment, MA2, MA2 with random operation sequence (RMA2) and MA4. MA2 with random operation sequence can increase the diversity of solutions. Two heuristics for operation sequence, OS1 and OS2, and two combination heuristics, MA2 + OS1 and MA2 + OS2, are also evaluated. The proposed four ensembles of heuristics are evaluated and compared to above simple heuristics and their combination. In initialization phase, job start time and machine start time are the same, 0. Hence, the problem is a general FJSP problem. We calculate the minimum value, average value and standard deviation obtained by each heuristic in 30 runs. The objectives are Makespan, Mworkload, Tworkload and E/T. The results for four objectives are shown in Tables 3–6.

Table 3 shows makespan results obtained by 11 compared heuristics for eight instances. It can be easily seen from Table 3 that RMA2, MA4, EH1, EH2, EH3 and EH4 heuristics (Group1) get

Table 2
The data of eight instances and job insertions.

Instance	Insertion order	Insertion time	Job number	Machine number	Number of operations
1	0	0	5	4	23
	1	10	1		4
	2	15	1		5
2	0	0	8	8	64
	1	12	1		8
	2	24	1		8
3	0	0	10	6	81
	1	15	1		8
	2	25	1		7
	3	35	1		6
4	0	0	10	10	100
	1	15	2		20
	2	30	2		20
5	0	0	15	8	171
	1	21	2		18
	2	47	2		17
6	0	0	15	10	185
	1	14	2		20
	2	31	2		19
	3	44	2		19
7	0	0	20	10	308
	1	21	3		38
	2	52	3		32
8	0	0	20	15	355
	1	18	3		42
	2	34	3		42
	3	49	2		25

Table 3
Makespan results by 11 heuristics for eight instances.

Instance/heuristics	Instance1			Instance2			Instance3			Instance4			Instance5			Instance6			Instance7			Instance8		
	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD
MA2	39	39	0	90	90	0	106	106	0	111	111	0	195	195	0	174	174	0	263	263	0	223	223	0
RMA2	28	34.9	4.4	53	62	6.6	75	85.2	7	62	76.5	7	104	114	5.8	95	103.4	7.7	113	134	7	106	117.5	6.8
MA4	26	32.3	3.6	47	53.6	5.3	70	80.7	5.2	61	68.3	4.7	96	102.8	5	82	91.5	5.7	111	123	5.7	97	106.3	5.4
OS1	42	56.3	9.9	75	92.2	8.9	108	126.6	10.2	98	118.2	10.2	179	196.8	12.5	157	177.1	13.2	231	263.7	20	225	242.2	9.1
OS2	40	52.7	8.1	70	89.2	9.7	102	122.7	12.6	101	118.5	11.3	169	187.3	13.4	149	175.9	11.5	238	263	14	212	236.9	11.7
MA2 + OS1	39	39	0	90	90	0	106	106	0	111	111	0	195	195	0	174	174	0	263	263	0	223	223	0
MA2 + OS2	46	46	0	95	95	0	103	103	0	107	107	0	179	179	0	151	151	0	261	261	0	223	223	0
EH1	29	34.2	3.4	52	58.3	5.4	69	80.6	6.5	58	71.3	6.6	101	113.3	5.8	97	109.8	5.7	127	137.4	6	114	121.3	5.8
EH2	27	30.2	3	50	58.4	4.4	69	77.7	5.3	59	70.2	5.6	95	109.4	6.6	87	94.8	4.1	124	132.4	4.4	100	110.7	5
EH3	26	31.1	4.5	42	48.7	3.3	63	71.7	5.5	50	56.2	3.3	89	96.2	4.6	83	91.1	4.8	106	115.7	4.7	97	103.8	4.3
EH4	28	28	0	51	51	0	77	77	0	67	67	0	92	92	0	82	82	0	117	117	0	100	100	0

Table 4
E/*T* results by 11 heuristics for eight instances.

Instance/heuristics	Instance1			Instance2			Instance3			Instance4			Instance5			Instance6			Instance7			Instance8			No. of v. in bold
	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	
MA2	**2**	**2.0**	0.0	22	22.0	0.0	23	23.0	0.0	30	30.0	0.0	72	72.0	0.0	56	56.0	0.0	106	106.0	0.0	69	69.0	0.0	2
RMA2	**3**	10.1	2.4	**7**	**11.5**	2.2	**5**	**11.1**	2.6	**9**	**15.1**	3.0	**16**	**19.5**	2.3	**14**	**21.0**	3.2	**21**	**28.9**	3.9	**36**	**45.8**	3.9	15
MA4	7	12.2	2.3	**10**	**15.9**	2.6	**8**	**11.8**	2.1	**14**	20.5	2.3	**18**	**21.1**	1.8	**22**	27.4	2.8	**28**	35.4	3.8	45	53.8	4.0	9
OS1	**2**	**8.3**	4.4	**8**	19.3	6.6	18	33.6	8.5	21	29.6	8.2	50	66.2	11.6	36	52.6	7.4	80	100.5	11.4	54	71.5	9.5	3
OS2	**3**	11.3	7.2	13	23.7	6.3	27	39.3	8.0	17	32.0	8.1	62	74.5	8.5	42	57.5	9.2	78	105.6	14.9	66	80.8	7.2	1
MA2 + OS1	**2**	**2.0**	0.0	22	22.0	0.0	23	23.0	0.0	30	30.0	0.0	72	72.0	0.0	56	56.0	0.0	106	106.0	0.0	69	69.0	0.0	2
MA2 + OS2	6	**6.0**	0.0	20	20.0	0.0	22	22.0	0.0	17	17.0	0.0	52	52.0	0.0	36	36.0	0.0	113	113.0	0.0	68	68.0	0.0	1
EH1	**2**	**8.2**	3.4	**9**	**11.1**	1.7	**8**	**12.7**	2.3	**8**	**12.6**	2.5	**17**	**20.8**	1.8	**14**	**16.2**	1.4	**19**	**26.2**	2.2	**21**	**31.0**	4.6	16
EH2	8	12.4	2.4	**8**	**11.5**	2.2	**7**	**10.4**	2.2	**12**	**17.0**	3.2	**18**	**21.0**	1.6	**19**	**23.1**	2.3	**24**	**27.3**	1.8	**35**	**43.1**	3.6	14
EH3	7	10.6	2.6	**10**	**15.1**	2.4	**7**	**12.0**	2.3	18	22.4	2.2	**19**	**21.2**	1.3	**16**	**21.1**	2.5	**27**	**32.6**	2.6	**38**	**44.3**	3.6	11
EH4	15	15.0	0.0	**14**	**14.0**	0.0	**9**	**9.0**	0.0	24	24.0	0.0	**23**	**23.0**	0.0	28	28.0	0.0	34	34.0	0.0	52	52.0	0.0	5

Table 5
Mworkload results by 11 heuristics for eight instances.

Instance/heuristics	Instance1			Instance2			Instance3			Instance4			Instance5			Instance6			Instance7			Instance8		
	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD
MA2	33	33.0	0.0	53	53.0	0.0	74	74.0	0.0	57	57.0	0.0	124	124.0	0.0	109	109.0	0.0	174	174.0	0.0	137	137.0	0.0
RMA2	21	24.5	1.5	31	33.8	2.3	54	58.7	3.5	36	42.6	3.5	71	76.0	3.3	61	65.8	2.7	88	92.5	3.4	65	68.8	2.4
MA4	21	24.9	1.9	29	34.3	2.1	55	62.3	4.6	38	44.8	3.0	74	78.9	3.1	60	66.6	3.5	92	97.3	3.6	65	69.7	2.4
OS1	36	49.0	7.1	53	71.1	11.1	83	107.4	11.6	73	90.8	11.7	133	162.9	15.7	119	141.1	16.9	186	212.9	13.5	159	188.2	16.1
OS2	33	46.7	9.0	52	69.8	13.9	75	104.4	15.7	69	90.7	13.3	134	161.1	15.9	122	148.9	22.2	186	215.7	18.9	152	183.6	12.6
MA2 + OS1	33	33.0	0.0	53	53.0	0.0	74	74.0	0.0	57	57.0	0.0	124	124.0	0.0	109	109.0	0.0	174	174.0	0.0	137	137.0	0.0
MA2 + OS2	33	33.0	0.0	53	53.0	0.0	74	74.0	0.0	57	57.0	0.0	126	126.0	0.0	109	109.0	0.0	177	177.0	0.0	137	137.0	0.0
EH1	22	25.7	2.3	31	34.5	2.3	55	60.3	3.6	39	45.0	4.5	71	77.6	4.1	60	65.1	2.6	87	92.9	3.4	65	69.0	2.5
EH2	23	26.2	2.5	29	34.3	2.3	55	60.3	3.9	36	43.1	4.4	70	76.6	3.2	61	65.5	3.5	88	92.3	2.3	66	69.9	2.8
EH3	22	23.4	1.6	29	32.6	2.1	52	57.8	2.7	36	37.7	1.2	71	76.5	2.1	60	62.7	2.1	88	93.8	2.0	66	68.3	1.6
EH4	24	24.0	0.0	34	34.0	0.0	59	59.0	0.0	43	43.0	0.0	78	78.0	0.0	65	65.0	0.0	102	102.0	0.0	71	71.0	0.0

Table 6
Tworkload results by 11 heuristics for eight instances.

Instance/heuristics	Instance1			Instance2			Instance3			Instance4			Instance5			Instance6			Instance7			Instance8		
	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD	Min	Ave	SD
MA2	120	120	0	371	371	0	424	424	0	543	543	0	921	921	0	1020	1020	0	1695	1695	0	1983	1983	0
RMA2	78	86	4	213	228	9	296	312	8	300	329	12	511	545	12	548	578	13	811	844	16	845	910	22
MA4	79	84	4	204	223	10	301	324	14	308	332	9	544	580	24	553	592	17	834	895	22	904	951	21
OS1	115	131	9	326	365	17	417	461	24	494	545	26	886	933	30	936	988	22	1530	1616	45	1842	1976	59
OS2	97	126	10	325	355	18	402	457	27	496	547	25	883	934	31	940	1010	38	1556	1638	40	1839	1965	49
MA2 + OS1	120	120	0	371	371	0	424	424	0	543	543	0	921	921	0	1020	1020	0	1695	1695	0	1983	1983	0
MA2 + OS2	120	120	0	371	371	0	418	418	0	543	543	0	937	937	0	1007	1007	0	1692	1692	0	1986	1986	0
EH1	77	87	7	214	229	9	297	316	10	308	329	11	516	553	15	561	585	13	816	851	18	895	931	17
EH2	79	88	6	200	229	11	300	318	8	311	328	9	523	557	17	565	584	12	818	853	18	905	930	14
EH3	77	82	6	212	226	8	302	330	15	314	331	8	549	592	16	562	593	15	857	914	20	941	974	16
EH4	80	80	0	218	218	0	325	325	0	329	329	0	601	601	0	614	614	0	924	924	0	991	991	0

similar minimum values, and average values for eight instances. For example, the minimum values and average values of instance 1 are {28,26,29,27,26,28} and {34.9,32.3.34.2,30.2,31.1,28}. The similar results can be found for other seven instances by these six heuristics. MA2, OS1, OS2, MA2 + OS1, and MA2 + OS2 heuristics (Group2) find similar minimum values and average values. For example, the minimum values and average values of instance 1 are {39,42,40,39,46} and {39,56.3,52.7,39,46}. The similar results can be found for other seven instances by these five heuristics. For eight instances, the minimum values and average values obtained by heuristics in Group1 are better than those obtained by heuristics in Group 2. The standard deviation of MA2, MA2 + OS1, MA2 + OS2, and EH4 are zero for eight instances because these four heuristics have similar features and get the same results in 30 runs. Expect these four heuristics, the heuristics in Group1 have better SD values than those obtained by the heuristics in Group2. For example, RMA2, MA4, EH1, EH2, and EH3 heuristics get the SD values {6.6,5.3,5.4,4.4,3.3} for instance 2 while the corresponding values by OS1 and OS2 are {8.9,9.7}. Similar results can be found for other seven instances. Hence, the six heuristics in Group1 have better min, ave and SD results than the five heuristics in Group2.

Table 4 shows E/T results of eight instances obtained by 11 heuristics. For example, the instance 2's min values are {7,10,8,9,8,10} by RMA2, MA4, OS1, EH1, EH2, and EH3. OS2 and EH4 obtain larger min values {13,14} for instance 2 while MA2, MA2 + OS1 and MA2 + OS2 get largest min values {22,22,20}. It is clear that RMA2, MA4, OS1, EH1, EH2, and EH3 heuristics have the better quality of min values and ave values. In Table 4, the min values and ave values with better and similar quality are set to "Bold" font. We count the number of values with "Bold" font for each heuristic. The results are shown in the last column of Table 4. The numbers in boldface by RMA2, EH1, EH2, and EH3 heuristics are more than 10. MA4 and EH4 heuristics get 9 and 5 "Bold" font values while MA2, OS1, OS2, MA2 + OS1, and MA2 + OS2 heuristics have less values with "Bold" font. For eight instances, the SD values by 11 heuristics have similar values with respect to makespan objective.

Tables 5 and 6 show the maximum machine workload and total machine workload of eight instances by 11 heuristics. Because both maximum machine workload and total machine workload are related to machine workload objective, Tables 5 and 6 have similar features. In these tables, RMA2, MA4, EH1, EH2, and EH3 heuristics, get similar quality min values and ave values for eights instances. These values are better than those obtained by MA2, OS1, OS2, MA2 + OS1, and MA2 + OS2 heuristics. EH4 is a special heuristic for maximum machine workload and total machine workload objectives. For small size instances, instance 1 and instance 2, EH4 can find similar min values and ave values as the RMA2, MA4, EH1, EH2, and EH3 heuristics. For example, the min Mworkload and Tworkload values by EH4 are {24,80}, and {34,218} for the first two instances. However, with the instance size becoming lager, the results by EH4 are larger than those by RMA2, MA4, EH1, EH2, and EH3 heuristics. Because EH4 has the same value in 30 runs, the min value and ave value for the same instance are the same. However, for instance 7 and instance 8, the ave values by EH4 are worse than those by RMA2, MA4, EH1, EH2, and EH3 heuristics. Hence, EH4 is effective for small size instance with machine workload related objectives. The RMA2, MA4, EH1, EH2, and EH3 heuristics have better results than MA2, OS1, OS2, MA2 + OS1, and MA2 + OS2 heuristics for small and large instances.

In summary, we compare 11heuristics for eight instances with different sizes. Makespan, E/T, Mworkload and Tworkload objectives are discussed respectively. Based on the above comparisons and discussions, RMA2, MA4, EH1, EH2, EH3 and EH4 heuristics can obtain similar makespan results that are better than those obtained by other compared heuristics. RMA2, EH1, EH2, EH3 heuristics get the highest quality results for E/T objective. The results by MA4 and EH4 are worse than those obtained RMA2, EH1, EH2, EH3 heuristics and better than other five heuristics. For maximum machine workload and total machine workload objectives, RMA2, MA4, EH1, EH2, and EH3 heuristics find satisfactory results. EH4 heuristics is just suitable for small size instances. According to these results of initializing by 11 heuristics, we select RMA2, MA4, EH1, EH2, EH3 and EH4 heuristics for rescheduling in Section 4.3. These six heuristics can get similar SD values for the same objective of the same instance. In Section 4.3, we will discuss these six heuristics in detail for rescheduling with new job insertion.

4.3. Rescheduling for new job insertion

For eight instances, re-scheduling is for new job(s) insertion. In each rescheduling, we use the previous scheduling solution with minimum results in 30 runs and run each algorithm for each instance 30 times. The minimum value and the average value in 30 runs are computed. To compare the six heuristics, we calculate the relative percentage increase (RPI) of minimum and average values by each heuristic. The formula is shown as follows:

$$\mathrm{RPI}(C_M^i) = \frac{C_M^i - C_M^*}{C_M^*} \times 100 \tag{5}$$

where C_M^i is the best min/ave objective value obtained by the ith heuristic in 30 runs, C_M^* is the best min/ave objective value found by six compared heuristics in Section 4.2. Obviously, the smaller the RPI value, the better result the heuristic finds. To show the algorithm performance, we also calculate the average relative percentage increase (ARPI) of each heuristics for eight instances. The results are shown in Tables 7–10 for makespan, E/T, Mworkload and Tworkload respectively. In these four tables, the first two columns show the instance number and new job(s) insertion order. Each heuristic includes two columns for min and ave RPI values. The last row is the min and ave APRI of each heuristic for eight instances.

Table 7 shows makespan rescheduling results for nineteen new job insertions. For RPI of min values, EH3 heuristic has eight results with value "0.00" while MA4 gets seven "0.00" values. That means EH3 and MA4 find eight minimum results and seven minimum results in nineteen new job insertion. The number of values "0.00" by EH4 heuristic and EH2 heuristic are five and one respectively. RMA2 and EH1 heuristics have no "0.00" value results. For all insertion rescheduling, the ARPI of min values by RMA2, MA4, EH1, EH2, EH3 and EH4, six heuristics are 14.20, 2.89, 15.45, 9.39, 1.89, and 8.29, respectively. EH3 obtains best min ARPI result. For ave RPI, EH4 have fourteen results with value "0.00" and much more than other five heuristics. The ARPI of ave value by EH4 is 2.78. This result is also the best one among the six compared heuristics. EH3 gets the second best ave value ARPI result with value 5.27 and the third one is MA4 with value 8.47. Hence, MA4, EH3 and EH4 heuristics have better performance than RMA2, EH1 and EH2 heuristics for makespan criterion.

Table 8 shows E/T rescheduling results for nineteen new job insertions in eight instances. For min RPI, RMA2 and MA4 get 12 results and 10 results with value "0.00". Compared to other four heuristics, RMA2 and MA4 heuristics have better ARPI results (16.46, 22.19). For ave RPI results, RMA2 obtains five results with value "0.00". EH1 to EH4 heuristics get three or four results with value "0.00" while MA4 has one. RMA2 and EH2 heuristics get better ARPI results (19.34, 19.37) than other four compared heuristics.

Table 7
Makespan rescheduling results by six heuristics for eight instances.

Instance	Insertion Order	RMA2		MA4		EH1		EH2		EH3		EH4	
		Min %	Ave %	Min %	Ave %	Min %	Ave %	Min %	Ave %	Min %	Ave %	Min %	Ave %
1	1	10.71	12.09	0.00	0.00	10.71	9.15	21.43	15.69	7.14	5.12	28.57	17.65
	2	3.03	6.72	0.00	0.00	6.06	8.06	6.06	2.42	3.03	9.77	33.33	18.28
2	1	17.02	16.55	10.64	6.18	19.15	17.64	21.28	16.73	0.00	3.33	17.02	0.00
	2	7.41	10.62	3.70	10.27	11.11	18.49	16.67	19.72	0.00	0.00	14.81	8.52
3	1	14.06	13.10	10.94	14.93	15.63	13.80	10.94	12.82	0.00	1.08	10.94	0.00
	2	7.69	18.25	2.56	14.50	12.82	21.38	0.00	8.50	1.28	7.50	2.56	0.00
	3	9.88	20.74	6.17	16.79	12.35	24.20	6.17	11.98	1.23	9.67	0.00	0.00
4	1	18.97	18.44	5.17	12.03	20.69	24.53	15.52	19.84	0.00	2.19	10.34	0.00
	2	13.51	20.77	0.00	8.36	12.16	20.00	9.46	13.09	0.00	0.00	13.51	7.46
5	1	13.86	24.26	0.00	10.20	10.89	21.09	8.91	14.75	5.94	13.40	0.00	0.00
	2	12.73	12.39	0.00	1.88	12.73	12.65	2.73	6.84	3.64	5.10	6.36	0.00
6	1	18.82	25.28	2.35	7.42	20.00	24.61	2.35	11.35	0.00	4.79	4.71	0.00
	2	15.73	26.74	2.25	15.62	19.10	30.67	6.74	14.16	3.37	10.11	0.00	0.00
	3	18.09	20.66	3.19	8.20	20.21	27.04	6.38	10.05	0.00	0.00	4.26	0.89
7	1	12.70	21.11	3.97	11.43	13.49	22.14	8.73	15.56	0.79	8.07	0.00	0.00
	2	11.76	21.99	2.94	8.68	12.50	20.51	5.88	12.21	2.21	6.27	0.00	0.00
8	1	18.00	26.50	1.00	6.70	19.00	26.80	8.00	17.10	0.00	7.23	0.00	0.00
	2	24.07	21.64	0.00	1.29	20.37	20.17	9.26	9.14	3.70	1.64	7.41	0.00
	3	21.82	23.60	0.00	6.40	24.55	25.44	11.82	14.12	3.64	4.80	3.64	0.00
ARPI		14.20	19.02	2.89	8.47	15.45	20.44	9.39	12.95	1.89	5.27	8.29	2.78

Table 8
E/*T* rescheduling results by six heuristics for eight instances.

Instance	Insertion	RMA2		MA4		EH1		EH2		EH3		EH4	
	Order	Min %	Ave %	Min %	Ave %	Min %	Ave %	Min %	Ave %	Min %	Ave %	Min %	Ave %
1	1	0.00	31.79	0.00	43.71	66.67	37.75	100.00	50.33	33.33	0.00	233.33	98.68
	2	100.00	47.86	0.00	18.57	50.00	22.86	50.00	34.29	50.00	0.00	150.00	7.14
2	1	75.00	7.02	0.00	26.32	75.00	9.12	50.00	0.00	125.00	40.35	250.00	47.37
	2	0.00	0.33	0.00	11.03	16.67	1.33	33.33	0.00	33.33	20.33	83.33	10.00
3	1	16.67	35.32	0.00	5.16	33.33	34.92	0.00	0.00	33.33	29.76	83.33	30.95
	2	75.00	56.30	0.00	21.48	200.00	84.81	50.00	13.33	125.00	31.11	125.00	0.00
	3	20.00	44.81	0.00	14.07	140.00	87.78	60.00	26.67	120.00	44.81	80.00	0.00
4	1	0.00	0.00	50.00	56.38	83.33	100.71	16.67	9.22	66.67	38.30	250.00	123.40
	2	0.00	2.42	50.00	24.85	66.67	10.30	33.33	9.70	66.67	6.06	83.33	0.00
5	1	0.00	0.00	0.00	6.80	28.57	32.04	7.14	8.74	14.29	6.21	42.86	16.50
	2	0.00	15.56	0.00	11.33	66.67	73.33	58.33	37.33	33.33	28.44	25.00	0.00
6	1	0.00	7.72	21.43	21.95	0.00	0.00	14.29	10.16	0.00	0.00	57.14	34.15
	2	0.00	4.18	0.00	14.39	16.67	22.51	16.67	11.83	8.33	0.00	66.67	39.21
	3	8.33	15.85	8.33	16.08	16.67	23.31	25.00	13.99	0.00	0.00	66.67	39.86
7	1	0.00	0.00	42.86	23.53	28.57	7.35	28.57	4.74	42.86	12.75	114.29	47.06
	2	0.00	0.00	12.50	9.26	37.50	27.61	25.00	8.59	31.25	9.43	43.75	16.16
8	1	17.65	61.49	100.00	101.90	0.00	0.00	35.29	58.16	70.59	68.62	188.24	132.96
	2	0.00	36.75	75.00	92.07	0.00	0.00	31.25	44.10	12.50	37.14	100.00	85.69
	3	0.00	0.00	61.54	67.14	15.38	3.02	23.08	26.81	15.38	11.09	146.15	93.55
ARPI		16.46	19.34	22.19	30.84	49.56	30.46	34.63	19.37	46.41	20.23	115.22	43.30

We can conclude that RMA2, MA4 and EH2 heuristics are more effective than the other three heuristics.

Tables 9 and 10 show the workload related rescheduling results. Mworkload results are shown in Table 9 while Tworkload results are shown in Table 10. We can see from Table 9 that MA4 heuristic obtains the most min RPI results with value "0.00" and the ARPI result, 3.68, is the best one among six heuristics. EH2 and EH3 heuristics have 5 results with value "0.00". The ARPI results of these two heuristics are 6.33 and 8.87 that are better than those by RMA2, EH1 and EH4 heuristics. For ave RPI, EH4 has the most results with value "0.00" and the ARPI results are the best one among all compared heuristics. MA2 and EH3 have similar ave RPI results and the ARPI results of MA2 and EH3 are 9.00 and 9.47 that are better than those by RMA2, EH1 and EH2 heuristics. It can be seen from Table 10 that MA4 gets most min RPI results with value "0.00" and the best ARPI result, 4.38, for all new job insertion rescheduling. EH4 has most ave RPI results with value "0.00" and the best ARPI result, 2.11. The ave ARPI result of MA4 is 6.26 while the min ARPI result by EH4 is 7.48. The two results are better than corresponding values by other four compared heuristics. Among the remaining four heuristics, EH2 has better min ARPI (7.86) and ave ARPI (9.46) results than those obtained by RMA2, EH1, and EH3. Hence, we can conclude that MA4, EH2 and EH4 three heuristics are better than RMA2, EH1 and EH3

Table 9
Mworkload rescheduling results by six heuristics for eight instances.

Instance	Insertion	RMA2		MA4		EH1		EH2		EH3		EH4	
	Order	Min %	Ave %	Min %	Ave %	Min %	Ave %	Min %	Ave %	Min %	Ave %	Min %	Ave %
1	1	12.50	20.78	0.00	15.49	0.00	6.27	6.25	13.53	0.00	4.51	6.25	0.00
	2	28.57	20.62	0.00	0.00	35.71	23.54	21.43	11.87	28.57	14.79	50.00	22.57
2	1	3.70	9.54	0.00	8.51	3.70	5.63	0.00	6.78	7.41	10.34	7.41	0.00
	2	20.83	28.54	4.17	15.27	12.50	19.15	0.00	7.76	0.00	0.00	16.67	5.13
3	1	2.27	8.94	0.00	7.02	4.55	10.35	0.00	7.87	4.55	7.73	6.82	0.00
	2	11.90	16.30	4.76	6.00	11.90	18.37	0.00	8.15	11.90	11.70	7.14	0.00
	3	12.82	19.84	0.00	4.47	0.00	8.05	2.56	8.21	12.82	14.23	5.13	0.00
4	1	16.13	20.70	16.13	27.40	6.45	12.40	6.45	21.20	0.00	0.00	19.35	11.00
	2	6.67	12.75	30.00	33.00	3.33	4.15	33.33	44.94	0.00	0.00	16.67	6.28
5	1	1.56	6.06	1.56	3.33	6.25	11.23	6.25	8.20	0.00	0.00	10.94	5.81
	2	14.00	20.45	0.00	6.79	20.00	26.41	10.00	14.74	0.00	0.71	4.00	0.00
6	1	1.82	10.18	3.64	12.73	7.27	16.12	1.82	11.09	5.45	10.18	0.00	0.00
	2	4.08	9.73	0.00	10.20	22.45	32.52	0.00	7.89	14.29	20.48	0.00	0.00
	3	12.20	24.80	4.88	11.46	39.02	54.72	2.44	14.23	24.39	31.30	0.00	0.00
7	1	0.00	0.00	4.71	4.97	8.24	7.98	4.71	3.60	12.94	11.14	17.65	11.36
	2	2.70	7.91	0.00	1.58	12.16	14.44	2.70	2.01	9.46	8.16	5.41	0.00
8	1	3.45	6.52	0.00	0.00	17.24	14.69	3.45	5.34	12.07	9.62	15.52	7.37
	2	12.73	13.90	0.00	0.00	25.45	28.36	10.91	9.55	12.73	9.21	10.91	3.39
	3	20.00	26.14	0.00	2.81	32.00	41.57	8.00	13.66	12.00	15.88	2.00	0.00
ARPI		9.89	14.93	3.68	9.00	14.12	18.73	6.33	11.61	8.87	9.47	10.62	3.84

Table 10
Tworkload rescheduling results by six heuristics for eight instances.

Instance	Insertion	RMA2		MA4		EH1		EH2		EH3		EH4	
	Order	Min %	Ave %	Min %	Ave %	Min %	Ave %	Min %	Ave %	Min %	Ave %	Min %	Ave %
1	1	16.33	23.08	28.57	33.14	18.37	23.08	0.00	2.56	8.16	14.68	6.12	0.00
	2	0.00	2.73	5.77	7.94	32.69	36.00	3.85	7.03	30.77	32.61	5.77	0.00
2	1	11.05	12.73	7.73	12.75	1.66	3.93	0.00	6.23	8.29	12.70	4.42	0.00
	2	30.66	33.56	24.09	28.21	19.71	24.64	0.00	3.98	18.25	19.26	5.84	0.00
3	1	6.58	6.64	0.00	1.26	7.41	6.02	13.99	12.29	0.41	0.00	8.23	0.37
	2	13.68	9.96	0.00	0.00	14.10	12.55	11.97	9.03	6.41	2.12	11.54	1.29
	3	16.10	13.00	0.00	0.00	22.44	19.93	20.98	19.01	15.61	13.18	9.76	0.07
4	1	12.32	18.70	5.80	11.90	8.70	14.50	1.09	8.66	2.17	8.93	0.00	0.00
	2	7.05	18.38	2.90	8.49	14.94	26.00	14.11	25.48	6.22	13.06	0.00	0.00
5	1	5.49	3.88	0.00	0.00	5.49	6.83	7.91	7.70	16.92	13.21	16.92	9.38
	2	10.39	13.82	0.00	0.00	18.82	18.27	11.80	11.19	17.42	17.35	10.67	3.58
6	1	1.01	2.99	4.46	3.69	10.55	10.25	0.00	0.00	14.20	12.44	8.72	3.25
	2	3.39	2.79	0.00	0.00	21.27	19.12	6.33	3.76	17.87	17.90	6.79	0.15
	3	13.14	23.06	2.58	8.00	30.15	37.84	4.90	12.77	20.36	26.49	0.00	0.00
7	1	0.00	0.00	1.25	1.91	7.75	6.50	5.13	5.68	7.75	8.88	9.38	4.90
	2	11.94	12.59	0.00	0.00	17.00	18.54	11.49	13.01	9.65	11.90	10.41	6.28
8	1	3.05	3.98	0.00	0.00	8.78	9.42	4.39	4.46	12.32	13.13	10.85	6.69
	2	7.93	4.65	0.00	0.00	20.43	15.91	13.44	8.87	20.30	16.50	11.83	4.12
	3	17.68	16.92	0.00	1.65	35.37	35.02	18.00	18.01	27.86	27.22	4.85	0.00
ARPI		9.88	11.76	4.38	6.26	16.61	18.12	7.86	9.46	13.73	14.82	7.48	2.11

heuristics for workload related criterion in new job insertion rescheduling.

In summary, we compared six heuristics for makespan, E/T, Mworkload and Tworkload criteria in nineteen new job insertion rescheduling. For each objective, we select two or three heuristics which get better results than those obtained by other compared heuristics. MA4 heuristics can be selected for four objectives. EH2 is in heuristics set for E/T, Mworkload and Tworkload objectives while EH4 is suitable for makespan, Mworkload and Tworkload objectives. EH3 and RMA2 heuristics are suitable for makespan and E/T objective, respectively. In addition, the running times of heuristics mentioned in this study is very low and can be ignored compared to the setup time and processing time in shop floor. This feature is suitable for the new job insertion and obtains rescheduling results in a limited time to continue processing in shop floor.

5. Conclusions

This paper modeled flexible job shop problem with new job insertion requirement from remanufacturing environment. Scheduling FJSP with new job insertion consists of two phases, initializing scheduling and rescheduling after new job(s) insertion. Four ensembles of heuristics are proposed for scheduling and

rescheduling FJSP with new job(s) insertion. The objectives include makespan, E/T, Mworkload and Tworkload. Eleven simple and ensembles of heuristics are discussed and compared for initializing scheduling for eight instances from a remanufacturing enterprise. Six heuristics with satisfactory results are selected for rescheduling for new job(s) insertion in the rescheduling phase. The six heuristics are discussed and compared on four objectives and the compared heuristics have different performances for different objectives. Among four ensembles of heuristics, CH4 is the best for makespan, Mworkload and Tworkload objectives while EH2 is the best for E/T objective. In addition, Ensembles of heuristics are suitable for FJSP with new job(s) insertion because the running time Ensembles is very low and can be ignored compared to the setup time and processing time in real shop floor. The selected Ensembles can obtain rescheduling results in time for continued processing in real shop floor.

References

Al-Hinai, N., & ElMekkawy, T. Y. (2011). Robust and stable flexible job shop scheduling with random machine breakdowns using hybrid genetic algorithm. *International Journal of Production Economics, 132*(2), 279–291.

Brandimarte, P. (1993). Routing and scheduling in a flexible job shop by tabu search. *Annals of Operations Research, 41*, 157–183.

Brucker, P., & Schlie, R. (1990). Job-shop scheduling with multi-purpose machines. *Computing, 45*(4), 369–375.

Calleja, G., & Pastor, R. (2014). A dispatching algorithm for flexible job shop scheduling with transfer batches: An industrial application. *Production Planning and Control, 25*(2), 93–109.

Daniel, V. R., & Guide, J. (2000). Production planning and control for remanufacturing: Industry practice and research needs. *Journal of Operations Management, 18*, 467–483.

Ferguson, M. (2009). The value of quality grading in remanufacturing. *Production and Operations Management, 18*, 300–314.

Gao, K. Z., Suganthan, P. N., & Pan, Q. K. (2014). Discrete harmony search algorithm for flexible job shop scheduling problem with multiple objectives. *Journal of Intelligent Manufacturing*. http://dx.doi.org/10.1007/s10845-014-0869-8.

Gao, J., Sun, L., & Gen, M. (2008). A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems. *Computers and Operations Research, 35*(9), 2892–2907.

Garey, M. R., Johnson, D. S., & Sethi, R. (1976). The complexity of flow hop and job shop scheduling. *Mathematics of Operations Research, 1*(2), 117–129.

Jain, A. S., & Meeran, S. (1998). Deterministic job-shop scheduling: Past, present and future. *European Journal of Operational Research, 113*(2), 390–434.

Li, J. Q., Pan, Q. K., & Gao, K. Z. (2011). Pareto-based discrete artificial bee colony algorithm for multi-objective flexible job shop scheduling problems. *International Journal of Advanced Manufacturing Technology, 55*(9–12), 1159–1169.

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

Li, J. Q., Pan, Q. K., Suganthan, P. N., & Chua, T. J. (2011). A hybrid tabu search algorithm with an efficient neighborhood structure for the flexible job shop scheduling problem. *International Journal of Advanced Manufacturing Technology, 52*(5–8), 683–697.

Mousakhani, M. (2013). Sequence-dependent setup time flexible job shop scheduling problem to minimize total tardiness. *International Journal of Production Research, 51*(12), 3476–3487.

Pezzella, F., Morganti, G., & Ciaschetti, G. (2008). A genetic algorithm for the flexible job-shop scheduling problem. *Computers & Operations Research, 35*(10), 3202–3212.

Vilcot, G., & Billaut, J. (2011). A tabu search algorithm for solving a multi-criteria flexible job shop scheduling problem. *International Journal of Production Research, 49*(23), 6963–6980.

Wang, L., Wang, S. Y., & Liu, M. (2013). A Pareto-based estimation of distribution algorithm for the multi-objective flexible job-shop scheduling problem. *International Journal of Production Research*. http://dx.doi.org/10.1080/00207543.2012.752588.

Wang, S. Y., Wang, L., Xu, Y., & Liu, M. (2013). An effective estimation of distribution algorithm for the flexible job shop scheduling problem with fuzzy processing time. *International Journal of Production Research, 51*(12), 3778–3793.

Wang, Y. M., Yin, H. L., & Qin, K. (2013). A novel genetic algorithm for flexible job shop scheduling problem with machine disruptions. *International Journal of Advanced Manufacturing Technology, 68*(5–8), 1317–1326.

Wang, L., Zhou, G., Xu, Y., & Liu, M. (2013). A hybrid artificial bee colony algorithm for the fuzzy flexible job shop scheduling problem. *International Journal of Production Research, 51*(12), 3593–3608.

Wang, L., Zhou, G., Xu, Y., Wang, S. Y., & Liu, M. (2012). An effective artificial bee colony algorithm for the flexible job shop scheduling problem. *International Journal of Advanced Manufacturing Technology, 60*(1–4), 303–315.

Xiong, J., Xing, L., & Chen, Y. W. (2013). Robust scheduling for multi-objective flexible job shop problems with random machine breakdowns. *International Journal of Production Economics, 141*(1), 112–126.

Yuan, Y., & Xu, H. (2013). An integrated search heuristic for large-scale flexible job shop scheduling problems. *Computers and Operations Research, 40*(12), 2864–2877.