A parallel search genetic algorithm for the flexible job shop scheduling problem with regular machine halt time

Guangcan Yang and Hegen Xiong

October 11, 2019

MISTA 2019

A parallel search genetic algorithm for the flexible job shop scheduling problem with regular machine halt time

Guangcan Yang • Hegen Xiong

Abstract This paper presents a more realistic flexible job shop scheduling problem, the flexible job shop scheduling problem with regular machine halt time (RMHT-FJSP). Different from machine breakdowns and maintenances, the regular machine halt time is deterministic, regular and frequent. The objective is to minimize the number of tardiness jobs (JTN), total tardiness time (TTT) and average machine idle time (AMIT). The objective of minimizing AMIT is to ensure operations on a machine close to each other, and it is only used as a non-important selection criterion in this paper. To enhance the optimization ability of genetic algorithm, serval kinds of crossover operators and mutation operators are adopted simultaneously, and the buffer population to integrate old population and new individuals generated by these operators during evolution process is proposed, we called this algorithm as parallel search genetic algorithm (PSGA). Further, a fitness function designed by pre-experiment is studied. A computational experiment is made. Comparisons between FJSP and RMHT-FJSP are also represented.

1 Introduction

The flexible job shop scheduling problem (FJSP), which is a kind of manufacturing system scheduling problem that is more practical than the job shop scheduling problem (JSP), has been widely studied [1-3]. Majority of related papers have been researched with machines can work continuously [4-6], and some have considered it with machine breakdowns and maintenances [7-10]. However, in the actual manufacturing system, due to rest time is required in daily production, sometimes even if the machines are available, the machines should stop operation during the rest time period. Unlike machine breakdowns and maintenances-related downtime [11-13], the working time with considering regular machine halt time is deterministic, regular and frequent. With this background, this paper presents the flexible job shop scheduling problem with regular machine halt time (RMHT-FJSP). In this paper, we describe the RMHT-FJSP. By

Guangcan Yang
Affiliation
E-mail: guangcanyang@yeah.net

Hegen Xiong
Affiliation
E-mail: xionghegen@126.com

a parallel search genetic algorithm (PSGA), experiments have been made, and the result shows that the problem can be effectively solved.

The remaining sections of this paper is organized as followings: In section 2, related researches are represented. Section 3 describes the RMHT-FJSP. Section 4 introduces the parallel search genetic algorithm. In section 5, experiments are made to verify the PSGA for RMHT-FJSP, and we adopt dynamic due date setting method. With the experiment results, discussion and conclusion are made in section 6.

2 Literature review

For JSP or FJSP in practical, some related works has been researched with considering various uncertain factors, such as machine breakdowns, rush orders [14], hot orders [15], preventive maintenances [16-18], delay arrival of a pre-arranged orders, cancellations of already handled jobs and changes in lot size [19], flexible workdays [20], etc. Machine breakdowns has been researched in [5], [21-31], etc. In these researches, machine breakdowns are unpredictable, stochastic. And during this period, the machines are unavailability. And some of them have mentioned reschedule when machine breakdowns occur. The proposed RMHT-FJSP in this paper is different from thses works. For RMHT-FJSP, the machines are available all the time, and there is also no reschedule. The machine halt time is deterministic, regular and frequent, which occurs due to the required rest time for workers in a day, a week, or a month.

The JSP, FJSP are NP hard problems, and in practice, often the close to optimization solutions is enough. Hence, for this kind of problem, heuristic algorithms [32-33], intelligent algorithms, and scheduling rules [34-35] are widely researched. Intelligent algorithms is a kind of effective method, they are often evolutionary algorithms, such as variable neighborhood search [36], NSGA-II and NRGA [37], particle swarm optimization [38-39], differential evolution [40-41], flower pollination algorithm [42-43], and genetic algorithm [21], [44-47]. The FJSP has two subproblems [42], [48], machine assignment problem and operation sequence problem. For the two subproblems separately, most genetic algorithms only have one crossover operator, one mutation operator, and a fixed scale population. In this paper, a parallel search genetic algorithm is proposed. Several crossover operators and mutation operators are integrated in the evolution process, and the children generated by these operators are inserted into the buffer population. The new population is selected from the buffer population. Obviously, the PSGA enriches the diversity of reproducing new individuals, and by alternately zooming in and out of the search space with certain a rule, it may have stronger optimization search ability.

3 Problem formulation

The FJSP can be divided into total FJSP and partial FJSP [42]. The objectives including minimize the maximal completion time, total cost, total tardiness, etc. [47], [49-50]. In This paper, the partial FJSP with regular machine halt time is studied.

Some assumptions for the RMHT-FJSP are listed:

1) All machines halt at the same time points and have the same halt time;
2) A machine can only process one operation at a time;
3) All jobs are released at time zero and independent from each other;
4) Setting up time and transportation time between operations are ignored;
5) Machines are available during production process;
6) A machine should stop once it finishes all operations on it;

7) A job is delivered once it is finished;
8) Every job has the same operation numbers;
9) Each job has a due date;
10) An operation must be processed consciously on a machine;
11) The processing time on a machine of a job are the same.

The FJSP is generally defined as follows. There are n jobs $J = \{J_1, J_2, \dots, J_n\}$, m machines $M = \{M_1, M_2, \dots, M_m\}$. Ecah job J_i has a set of operations $O_i = \{O_{i1}, O_{i2}, \dots, O_{ij}\}$ that ordered by a fixed sequence. O_{ijk} represents O_{ij} is processed on M_k. The processing time of operation O_{ij} on machine M_k is represented as P_{ijk}. Let S_{ijk}, F_{ijk} be the start time and finish time of O_{ij} on M_k separately, and $C_i = Max\left(F_{ijk}\right), D_i$ be the completion time and due date of J_i separately. In the RMHT-FJSP, two sets with the same size $H = \left\{H_1, H_2, \dots, H_p\right\}, T = \{T_1, T_2, \dots, T_p\}$ are represented as the machine halt time points and the corresponding halt time.

As machine halt exists, i.e., the rest time which is pre-determined but also changeable exists. During the machine halt period (MHP), even if the machine is available, it should stop operations. An operation is also not allowed to start or be processed during MHP. And if an operation can be processed on a machine discontinuously, i.e., the operation time is separable, as the new completion time equals the old completion time plus the machine halt time, then the RMHT-FJSP can be easily handled with already researches.

But if an operation must be processed on a machine continuously, i.e., if an operation could start but can't complete before machine halt time point, then the start time should be changed till the operation can complete continuously and it should also satisfy the fixed operation sequence constrain. And this kind of problem has not been researched. Hence, this paper considers the case that operations must be processed on a machine continuously. And the S_{ijk} may be redefined by (1).

$$S'_{ijk} = H_t + T_t, if\ H_t \le S_{ijk} < H_t + T_t\ or\ H_t < F_{ijk} \le H_t + T_t, t = 1, 2, \dots, p. \quad (1)$$

The number of tardiness jobs (JTN) is calculated by (2).

$$JTN = \sum_1^n \begin{cases} 1, if\ \ C_i > D_i \\ 0, if\ C_i \le\ D_i \end{cases}, i = 1, 2, \dots, n. \quad (2)$$

The total tardiness time (TTT) is defined by (3).

$$TTT = \sum_1^n Max\{C_i - D_i, 0\}, i = 1, 2, \dots, n. \quad (3)$$

The average machine idle time (AMIT) is represented by (4).

$$AMIT = \frac{Max(F_{ijk}) - \sum_1^i \sum_1^j P_{ijk}}{m}, k = 1, 2, \dots, m. \quad (4)$$

The objective is formulated by (5), where $\sum_1^3 \mu_q = 1, q = 1, 2, 3$.

$$Min\ Objective = \mu_1 \cdot JTN + \mu_2 \cdot TTT + \ \mu_3 \cdot AMIT \quad (5)$$

4 The parallel search genetic algorithm

The genetic algorithm has global search ability, and is widely researched for specific problem types. Usually it starts with a random set of solutions called population, and each individual called chromosome represents a feasible solution. In this study, chromosome coding methods for machine assignment problem (MAP) and operation sequence problem (OSP) are presented, crossover operators and mutation operations are introduced. The proposed buffer population is to integrate offspring produced by these evolution operators. Taking an example with 3 jobs, 3 operations, and 5 machines to introduce the chromosome and evolution operators. The fixed operation sequences of each job are $OSJ_1 = \{2, 1, 3\}, OSJ_2 = \{1, 3, 2\}, OSJ_3 = \{3, 1, 2\}$. The machine sets for each operation are $MSO_1 = \{1, 5\}, MSO_2 = \{2, 3\}, MSO_3 = \{4, 5\}$. For example, the 1^{st} number in OSJ_1 is 2, it means that $O_{1,1}$ can be process on machine sets MSO_2, i.e., machine 2 and machine 3.

4.1 Chromosome representation

Since there are two subproblems in FJSP, two chromosomes are presented. A $n \times n_i$ matrix called MAC for machine assignment problem is shown in Figure 1, where n_i is the operation numbers of J_i. For example, row 1 and column 1 is 3, which means M_3 is assigned to process $O_{1,1}$ (the 1st operation in OSJ_1), and the meaning can be represented by $O_{1,1,3}$.

$$\text{MAC} = \begin{bmatrix} 3 & 5 & 5 \\ 1 & 4 & 2 \\ 4 & 5 & 2 \end{bmatrix}$$

Figure 1 The chromosome structure for machine assignment problem

An operation-based chromosome in [51] for operation sequence problem is also feasible for RMHT-FJSP. The chromosome called OSC has $n \times n_i$ genes, each gene contains information like job number, operation number, start processing time, etc. For example, a chromosome and the explanation for 3 jobs and 3 operations FJSP problem are shown in Figure 2.

OSC	1	3	2	1	3	1	2	2	3
Explanation	$O_{1,1}$	$O_{3,1}$	$O_{2,1}$	$O_{1,2}$	$O_{3,2}$	$O_{1,3}$	$O_{2,2}$	$O_{2,3}$	$O_{3,3}$

Figure 2 The chromosome for operation sequence problem

4.2 Fitness function

The objective in this paper is to minimize the JTN, TTT and $AMIT$. When the $JTN = 0$, then the $TTT = 0$, too. And in practice production scheduling, usually the $AMIT > 0$ and far larger than JTN. So, the objective function needs to be modified to get better performance during evolution process. Hence, a fitness function designed by (6) is adopted, where $OBJ_1 = JTN, OBJ_2 = TTT, OBJ_3 = AMIT$. With $\mu_3 < 0.1$ and $AMIT < 110$, it can be inferred that the $Fitness > 0.9$ when $JTN = 0$. In this paper, three kinds of fitness functions parameters are considered, FP = $\{(0.68, 0.22, 0.10), (0.71, 0.21, 0.08), (0.74, 0.20\ 0.06)\}$. And with Objectives = $\{(0, 0, 110), (1, 10, 140), (2, 30, 170), (3, 50, 200), (4, 70, 230), (5, 100, 260)\}$, the fitness value for each FP are illustrated in Figure 3. It shows that the designed fitness function can performance well.

$$Fitness = \frac{1}{1 + \sum_{q=1}^{3} \mu_q \cdot log10(1 + \mu_q \cdot OBJ_q)}, q = 1,2,3. \tag{6}$$

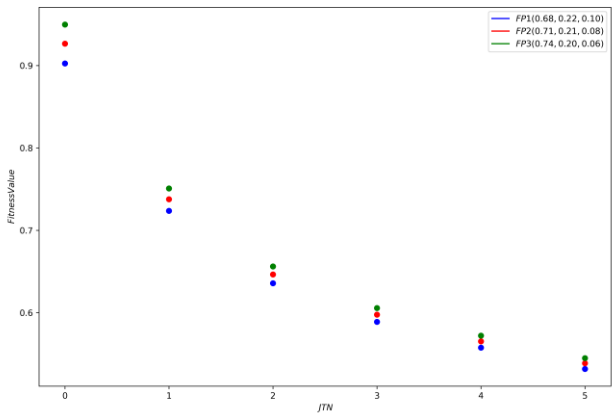

Figure 3 Pre-experiment of the fitness function parameters

4.3 Selection strategies

For different population types named initial population (new population) and buffer population, two selection strategies are considered. A roulette selection strategy [52] is adopted for initial population or new population, which the individuals selected are for crossover operations or mutation operations. For buffer population, a strategy that select the best ones or randomly select the worst ones by keeping the best one with a fixed size as the new population is proposed.

4.4 Crossover operators

There are two crossover operators for MAC, matched and random points exchange (MRPE), matched and rearrange (MR). And three operators for OSC, POX [51], parts exchange I (PEX1), parts exchange II (PEX2).

Given MAC1 and MAC2, crossover applied MRPE generates new MAC1 and new MAC2, by following procedures:

1) Randomly choose an operation, i.e., the same number in $OSJ_i, i = 1,2, \dots, n$;

2) With O_{ijk}, where the j^{th} number in OSJ_i is the column index in MAC, find the positions of machines in MAC1 and MAC2;

3) Exchange some elements found by procedure 2 between MAC1 and MAC2.

Different from MPRE, MR only needs one MAC, and just rearrange the elements found in procedure 2 of MRPE. Separately an example of MRPE and MR is illustrated in Figure 4 and Figure 5.

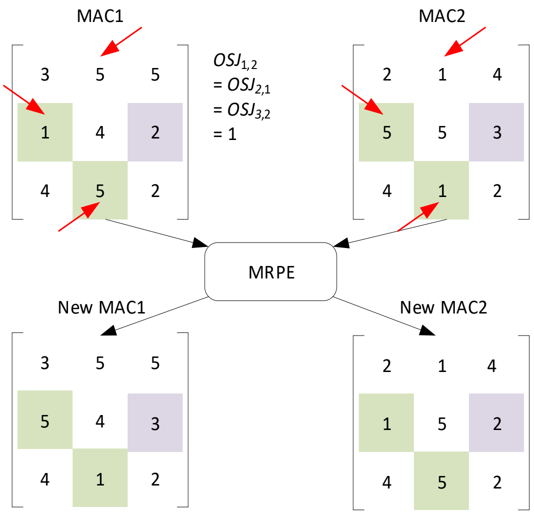

Figure 4 An example of MRPE

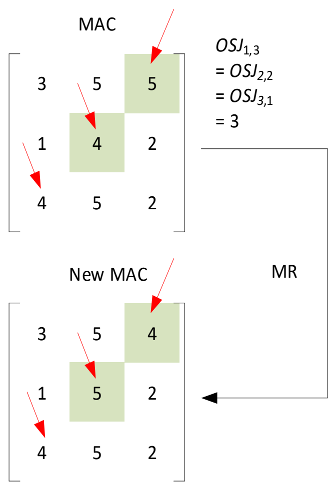

Figure 5 An example of MR

An example of POX is shown in Figure 6.

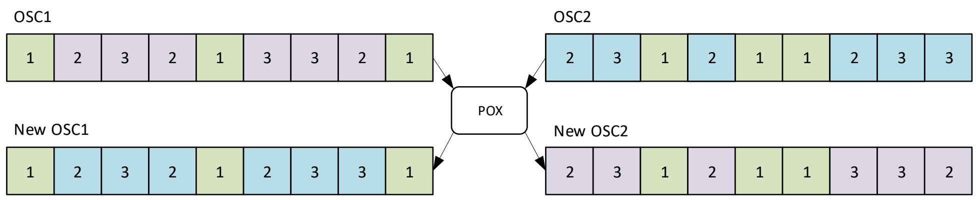

Figure 6 An example of POX

PEX1 and PEX2 has little differences. The PEX1 chooses four different points randomly in OSC, and exchange the genes between point 1 to point 2 and point 3 to point 4. The PEX2 only choose two different points randomly in OSC, and preserves the genes from point 1 to point 2, then exchange the genes of the other two parts. Also, an example of PEX1 and PEX2 is separately represented in Figure 7 and Figure 8.

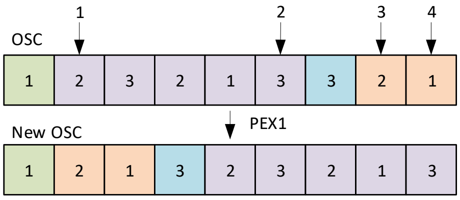

Figure 7 An example of PEX1

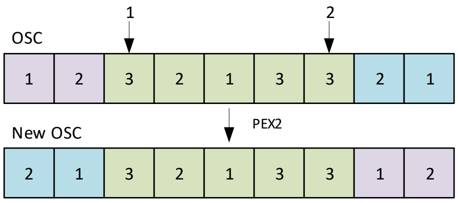

Figure 8 An example of PEX2

4.5 Mutation operators

A mutation operator for MAC, named random points replace (RPR) is similar to MRPE and MR. The first two procedures are the same. In procedure 3, the RPR replaces some elements found by procedure 2 from MSO_j. Figure 9 is an example of RPR.

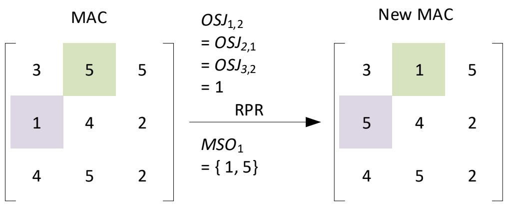

Figure 9 An example of RPR

Also, two mutation operators similar to [21], named parts reverse (PR) and discrete points reverse (DPR) are introduced. Like PEX2, PR chooses two different points randomly in OSC. Let the genes between the two points named part 1 and the other genes named part 2. If the genes in part 1 are no less than the genes in part 2, then reverse the genes in part 1, otherwise reverse the genes in part 2. And different from PR, DPR compares the number of selected genes and left genes. An example of PR and DPR is represented separately in Figure 10 and Figure 11.

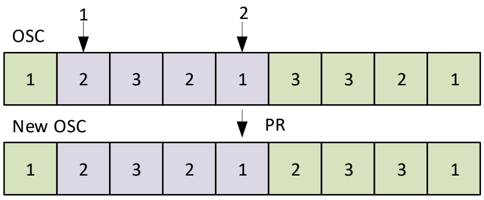

Figure 10 An example of PR

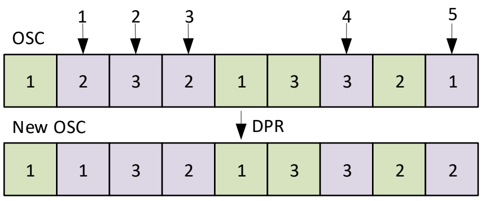

Figure 11 An example of DPR

Terminate condition for designed algorithm is that the $Fitness > 0.9$ and all the objectives have no changes in recent 3 iterations. The flow of PSGA is shown in Figure 12.

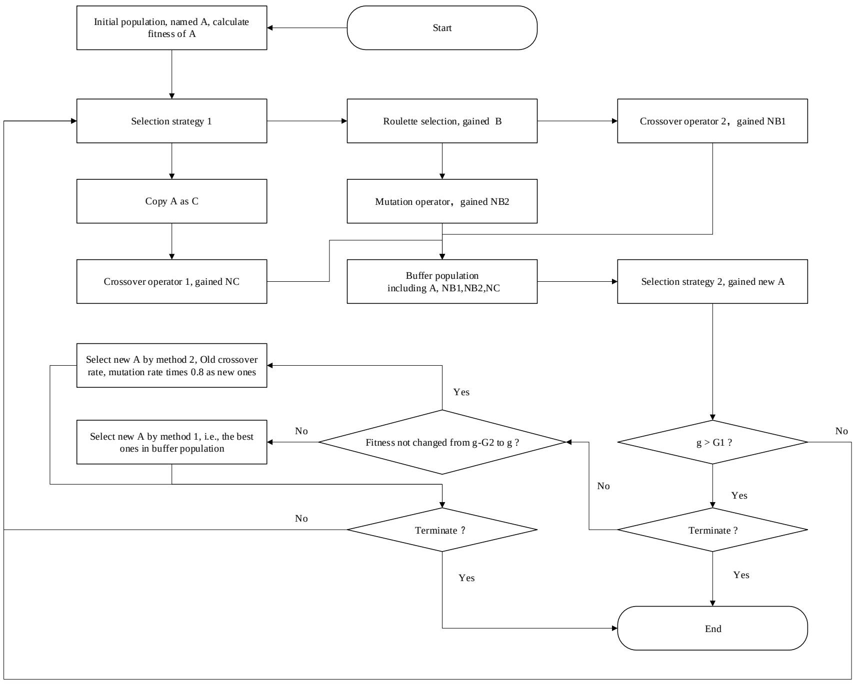

Figure 12 The flow of PSGA

5 Computational experiment

To verify the parallel search genetic algorithm, this part has made a computational experiment with 10 jobs, each job has 6 procedures, and 8 machines. Table 1 is the operation sequences. Table 2 is the processing time. The machine sets are $MSO_1 = \{1\}, MSO_2 = \{2,7\}, MSO_3 = \{3,8\}, MSO_4 = \{4\}, MSO_5 = \{5\}, MSO_6 = \{6,8\}$. In Table 1, e.g., row 2 column 1 is 2, it means that $O_{2,1}$ can be processed on machine sets MSO_2, i.e., machine 2 and machine 7. In Table 2, e.g., row 2 and column 1 is 3, it means that there needs 3 units time for J_2 on M_1.

A due date setting method is introduced in [53], and this paper uses (7).

$$D_i = c \times \sum_{j=1}^{n_i} P_{ijk}, k = 1,2,\dots,m, \text{ where } c \in \{2.5, 2.7, 2.8\} \quad (7)$$

Specially, machine halt time points and halt time separately is $H = \{30, 60, 90\}, T = \{2, 2, 2\}$. For example, $H1 = 30$, $T1 = 2$, it means that at time point 30, all machines should stop operations for 2 units time.

Table 1 The operation sequences

Job	Operation sequence					
1	1	4	5	2	6	3
2	2	3	6	1	4	5
3	3	6	4	2	1	5
4	6	4	3	5	1	2
5	1	4	2	6	5	3
6	6	4	1	5	2	3
7	3	1	5	4	2	6
8	6	5	4	2	1	3
9	5	1	2	6	3	4
10	1	5	2	6	3	4

Table 2 The processing time

Job/Machine	1	2	3	4	5	6	7	8
1	5	11	11	4	4	10	3	7
2	3	10	3	10	11	5	12	7
3	9	11	3	9	11	8	10	3
4	6	6	5	10	4	2	10	7
5	12	10	11	11	10	2	6	10
6	12	8	7	10	5	5	6	6
7	3	10	7	8	2	10	10	8
8	9	9	6	7	8	9	11	7
9	3	4	6	12	5	9	7	5
10	5	10	12	8	5	4	5	8

Experiments with basic GA, PSGA are made. The parameters for algorithm are in Table 3. And this paper creates special initial population, the JTN is between 20% - 80% of the total jobs. Programed with Python 3.7, and running 30 times on windows 10 with 1.5GHz and 4GB RAM, for per designed experiment, Table 4 and Table 5 show the results, where $\overline{Runtime}$ is the average runtime (seconds), Opt% is the percentage for $JTN = 0$, [b]Value means best, [w]Value means worst.

Table 3 Parameters for algorithm

Algorithm/Parameters	Population scale	Maximal generation	Crossover rate 1	Crossover rate 2	Mutation rate
GA	80	50	0.8	-	0.15
PSGA	40	50	0.8	0.75	0.15

Experiments with basic GA, roulette selection, MRPE, POX, RPR and PR are adopted. The population scale is 80, the maximal generation is 50, crossover rate is 0.8, mutation rate is 0.15. The results are shown in Table 4.

Table 4 Experiment results (GA)

Experiment	FP	c	$\overline{JTN}$	$\overline{TTT}$	$\overline{AMIT}$	$\overline{Fitness}$	$\overline{Runtime}$	Opt%
1	1	2.5	0.33 [b]0 [w]1	0.90 [b]0 [w]5	40.50 [b]33.88 [w]51.38	0.8866	5610.35	66.67%
2	2	2.5	0.30 [b]0 [w]1	0.80 [b]0 [w]6	39.99 [b]33.12 [w]44.62	0.9061	5611.13	70.00%
3	3	2.5	0.47 [b]0 [w]1	2.20 [b]0 [w]10	41.00 [b]31.12 [w]49.75	0.8870	5608.97	53.33%
4	1	2.7	0	0	35.37 [b]25.25 [w]42.75	0.9385	5612.30	100.00%
5	2	2.7	0	0	36.03 [b]31.12 [w]45.12	0.9551	5604.64	100.00%
6	3	2.7	0	0	35.09 [b]29.38 [w]41.88	0.9714	5622.56	100.00%
7	1	2.8	0	0	34.61 [b]31.62 [w]39.75	0.9391	5612.72	100.00%
8	2	2.8	0	0	34.65 [b]29.12 [w]39.38	0..9560	5611.92	100.00%
9	3	2.8	0	0	34.46 [b]25.62 [w]41.12	0.9717	5614.79	100.00%

Experiments with PSGA, the population scale is 40, the maximal generation is 50, crossover rate for MRPE, POX is 0.8, crossover rate for MR, PEX1 and PEX2 is 0.75, and mutation rate is 0.15. The results are shown in Table 5.

Table 5 Experiment results (PSGA)

Experiment	FP	c	$\overline{JTN}$	$\overline{TTT}$	$\overline{AMIT}$	$\overline{Fitness}$	$\overline{Runtime}$	Opt%
1	1	2.5	0.10 [b]0 [w]1	0.30 [b]0 [w]5	41.42 [b]37.12 [w]50.50	0.9191	5965.80	90.00%
2	2	2.5	0.07 [b]0 [w]1	0.20 [b]0 [w]3	41.84 [b]34.25 [w]54.88	0.9411	5755.23	93.33%
3	3	2.5	0.27 [b]0 [w]1	0.50 [b]0 [w]3	42.15 [b]31.25 [w]51.50	0.9254	6001.78	73.33%
4	1	2.7	0	0	43.02 [b]36.62 [w]51.00	0.9326	2625.16	100.00%
5	2	2.7	0	0	44.51 [b]31.50 [w]55.88	0.9501	2747.22	100.00%
6	3	2.7	0	0	45.43 [b]36.50 [w]57.88	0.9670	2541.22	100.00%
7	1	2.8	0	0	47.35 [b]37.88 [w]57.88	0.9297	1472.14	100.00%
8	2	2.8	0	0	46.46 [b]36.75 [w]54.62	0.9490	2007.22	100.00%
9	3	2.8	0	0	42.89 [b]37.38 [w]51.62	0.9679	1994.42	100.00%

The experiments results show that both GA and PSGA can find the approximate optimization solutions. When $c = 2.5$, PSGA is more stable than GA. And when $c = 2.7\ or\ 2.8$, PSGA costs less time than GA. During evolution, the fitness function may not work well sometime, e.g., FP1 = (0.68, 0.22, 0.06), when objectives are (3, 69, 54.75) and (4, 19, 56.5), where (3, 69, 54.75) is better, but the fitness value is smaller. But it also can find a better one. So, in practice, the larger μ_1 and smaller μ_3 may get better performance.

Take a solution by PSGA when $c = 2.5$ and $JTN = 1, TTT = 2, AMIT = 41.75$, the Gannt Chart for RMHT-FJSP with $H = \{30, 60, 90\}, T = \{2, 2, 2\}$ is shown in Figure 13, the Gannt Chart for FJSP, i.e., with $H = \{30, 60, 90\}, T = \{0, 0, 0\}$ is shown in Figure 14. In Figure 13, J_9 is tardiness, but in Figure 14, $C_9(91) < D_9\,(102)$.

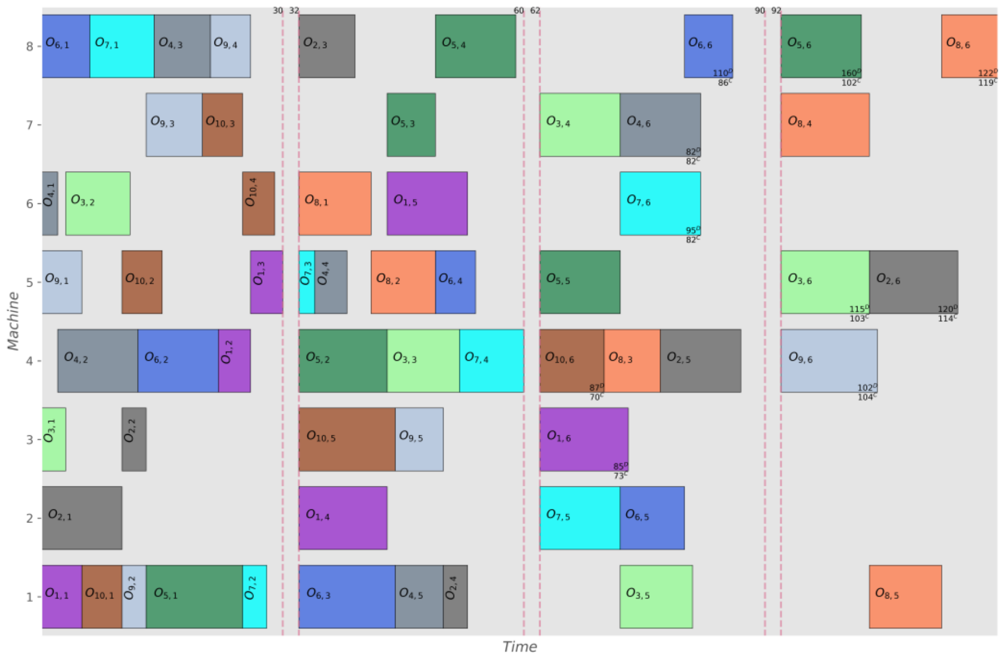

Figure 13 The Gannt Chart for RMHT-FJSP

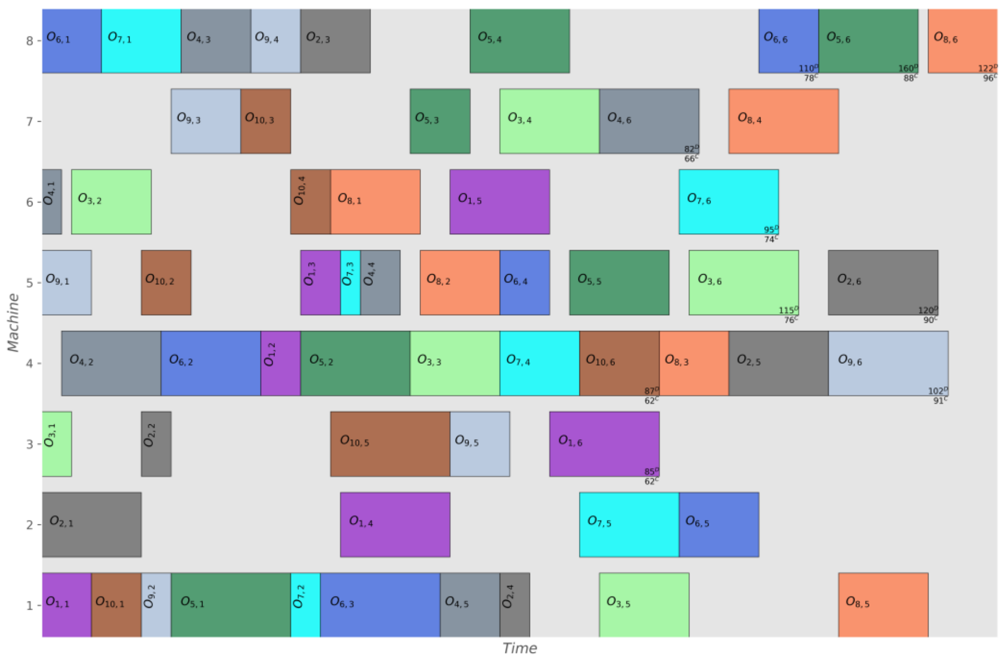

Figure 14 The Gannt Chart for FJSP

And take a solution by PSGA when $c = 2.5$ and $JTN = 0, TTT = 0, AMIT = 37.12$, the Gannt Chart for RMHT-FJSP with $H = \{30, 60, 90\}, T = \{2, 2, 2\}$ is shown in Figure 15, the Gannt Chart for FJSP, i.e., with $H = \{30, 60, 90\}, T = \{0, 0, 0\}$ is shown in Figure 16. Both in Figure 15 and Figure 16, all jobs complete before their due date. And separately the objective value trace curve, fitness value trace curve is illustrated in Figure 17 and Figure 18, it shows that the PSGA has strong optimization ability.

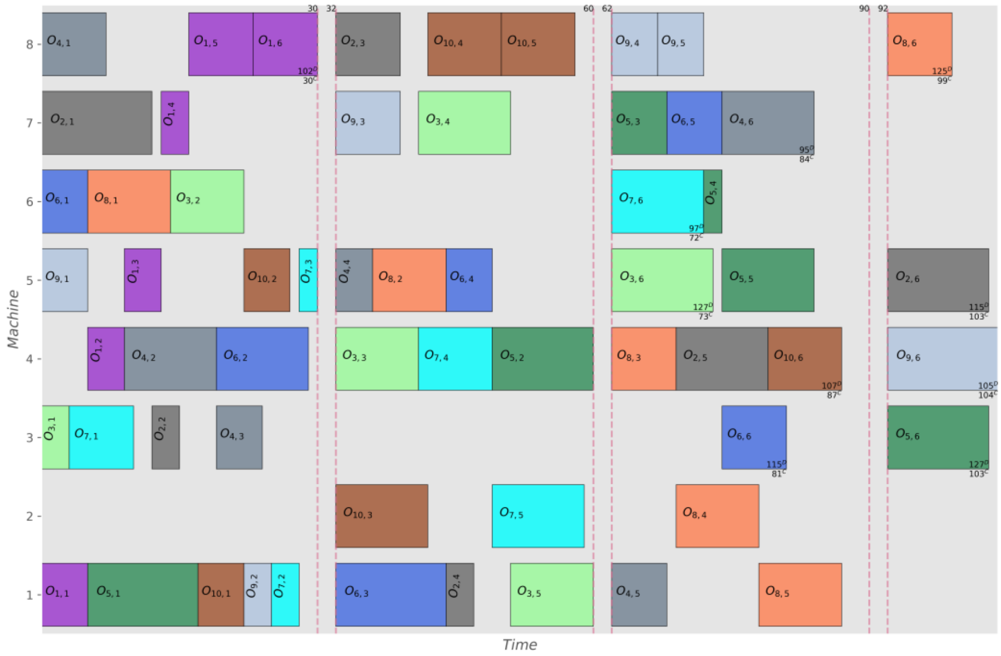

Figure 15 The Gannt Chart for RMHT-FJSP

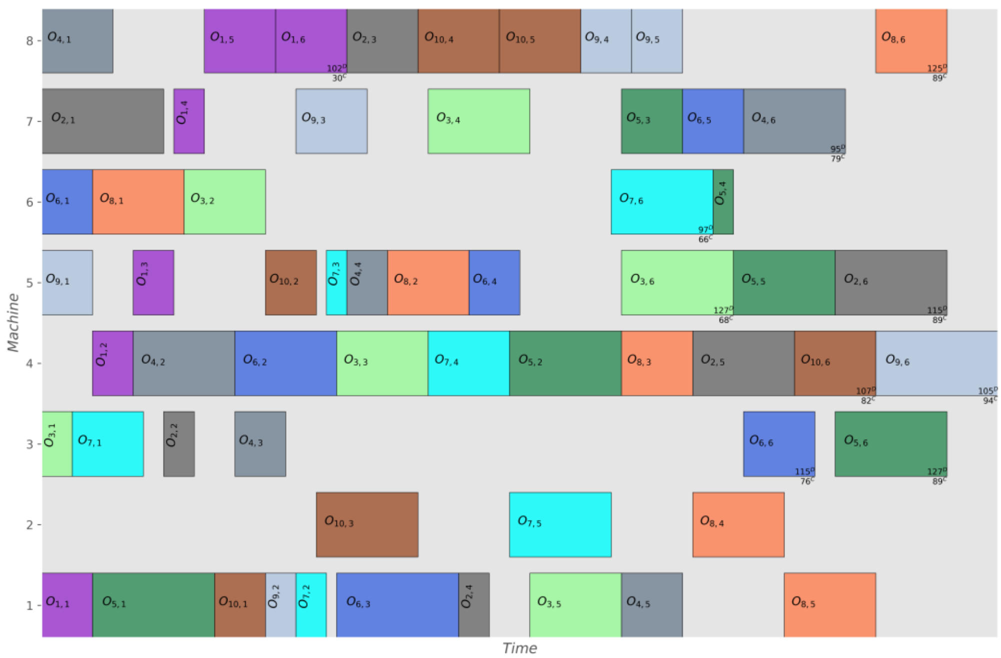

Figure 16 The Gannt Chart for FJSP

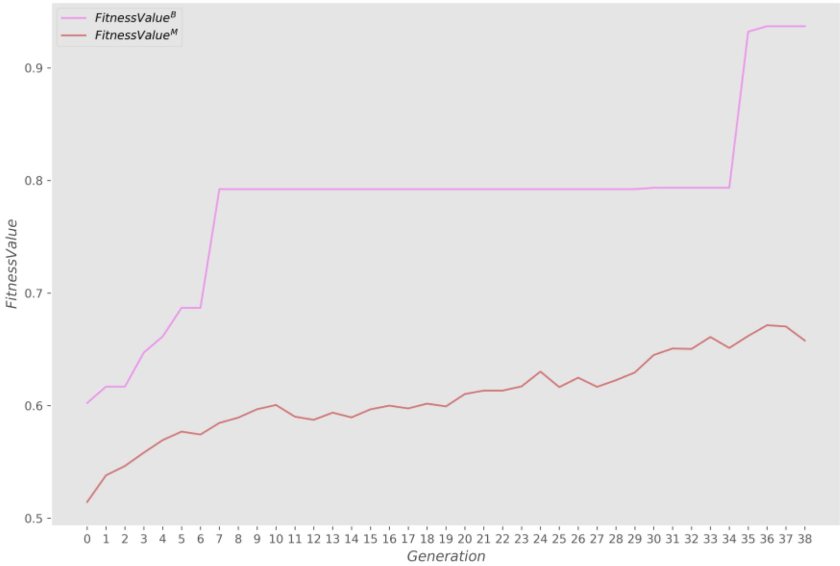

Figure 17 Fitness value

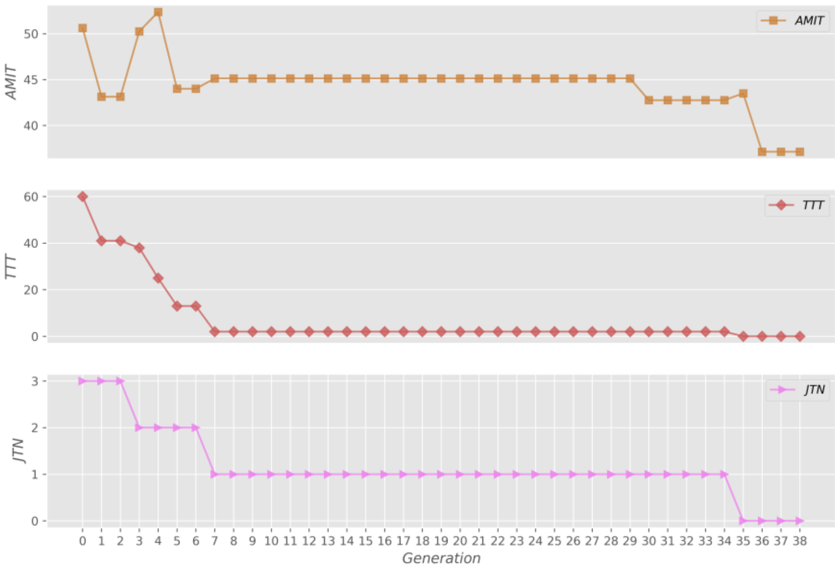

Figure 18 Objective value

In Figure 17, $FitnessValue^B$ is the best fitness value till current generation, and $FitnessValue^M$ is the mean fitness value of current generation. It shows that not only the $FitnessValue^B$ gets better and better, but also the $FitnessValue^M$ gets better and better. In figure 18, $JTN, TTT, AMIT$ changings are shown. When JTN or TTT gets better, the $AMIT$ may get worse than before, but in the end, all objectives can convergent to a better value.

6 Discussion and conclusion

In this paper, the flexible job shop scheduling problem with regular machine halt time has been researched. The objective is to minimize the JTN, TTT and $AMIT$. The PSGA, with buffer population that integrates serval kinds of crossover operators and mutation operators is introduced. A computational experiment has been made. The experiments results show that the RMHT-FJSP can be effectively solved in polynomials time by PSGA. Comparison between GA and PSGA shows PSGA is more stable and has stronger optimization ability. Further, by comparing the Gannt Chart in Figure 13 and Figure 14 or in Figure 15 and Figure 16, it can be seen that the RMHT-FJSP may be more reasonable in practice production.

Acknowledgements This work is supported by the National Natural Science Foundation of China under Grant No.51875422.

References

1. Ozguven, C., Y. Yavuz, and L. Ozbakir, *Mixed integer goal programming models for the flexible job-shop scheduling problems with separable and non-separable sequence dependent setup times.* Applied Mathematical Modelling, 2012. **36**(2): p. 846-858.
2. Lu, C., et al., An effective multi-objective discrete virus optimization algorithm for flexible job-shop scheduling problem with controllable processing times. Computers & Industrial Engineering, 2017. **104**: p. 156-174.
3. Zhou, Y., J.-J. Yang, and L.-Y. Zheng, *Multi-Agent Based Hyper-Heuristics for Multi-Objective Flexible Job Shop Scheduling: A Case Study in an Aero-Engine Blade Manufacturing Plant.* Ieee Access, 2019. **7**: p. 21147-21176.
4. Kolen, A.W.J., et al., *Interval scheduling: A survey.* Naval Research Logistics, 2007. **54**(5): p. 530-543.
5. Luo, H., et al., *Two-stage hybrid batching flowshop scheduling with blocking and machine availability constraints using genetic algorithm.* Robotics and Computer-Integrated Manufacturing, 2009. **25**(6): p. 962-971.
6. Xuan, H. and B. Li, *Scheduling dynamic hybrid flowshop with serial batching machines.* Journal of the Operational Research Society, 2013. **64**(6): p. 825-832.
7. Wu, C.-C. and W.-C. Lee, *A note on single-machine scheduling with learning effect and an availability constraint.* International Journal of Advanced Manufacturing Technology, 2007. **33**(5-6): p. 540-544.
8. Engin, O., et al., *Solving Fuzzy Job Shop Scheduling Problems with Availability Constraints Using a Scatter Search Method.* Journal of Multiple-Valued Logic and Soft Computing, 2013. **21**(3-4): p. 317-334.
9. Li, J.-Q., Q.-K. Pan, and M.F. Tasgetiren, *A discrete artificial bee colony algorithm for the multi-objective flexible job-shop scheduling problem with maintenance activities.* Applied Mathematical Modelling, 2014. **38**(3): p. 1111-1132.
10. Zandieh, M., A.R. Khatami, and S.H.A. Rahmati, *Flexible job shop scheduling under condition-based maintenance: Improved version of imperialist competitive algorithm.* Applied Soft Computing, 2017. **58**: p. 449-464.
11. Cheng, C., H. Tang, and C. Zhao, *Scheduling jobs on a machine subject to stochastic breakdowns to minimize absolute early-tardy penalties.* Science in China Series a-Mathematics, 2008. **51**(5): p. 864-888.
12. Tiemessen, H.G.H. and G.J. van Houtum, *Reducing costs of repairable inventory supply systems via dynamic scheduling.* International Journal of Production Economics, 2013. **143**(2): p. 478-488.
13. Wu, Z., S. Sun, and S. Xiao, *Risk measure of job shop scheduling with random machine breakdowns.* Computers & Operations Research, 2018. **99**: p. 1-12.

14. Moratori, P., S. Petrovic, and J.A. Vazquez-Rodriguez, *Integrating rush orders into existent schedules for a complex job shop problem.* Applied Intelligence, 2010. **32**(2): p. 205-215.
15. Zhao, F., et al., *A hybrid particle swarm optimisation algorithm and fuzzy logic for process planning and production scheduling integration in holonic manufacturing systems.* International Journal of Computer Integrated Manufacturing, 2010. **23**(1): p. 20-39.
16. Safari, E. and S.J. Sadjadi, *A hybrid method for flowshops scheduling with condition-based maintenance constraint and machines breakdown.* Expert Systems with Applications, 2011. **38**(3): p. 2020-2029.
17. Lu, Z., W. Cui, and X. Han, *Integrated production and preventive maintenance scheduling for a single machine with failure uncertainty.* Computers & Industrial Engineering, 2015. **80**: p. 236-244.
18. Li, R. and H. Ma, *Integrating Preventive Maintenance Planning and Production Scheduling under Reentrant Job Shop.* Mathematical Problems in Engineering, 2017.
19. Baykasoglu, A. and F.B. Ozsoydan, *Dynamic scheduling of parallel heat treatment furnaces: A case study at a manufacturing system.* Journal of Manufacturing Systems, 2018. **46**: p. 152-162.
20. Zhang, J., J. Yang, and Y. Zhou, *Robust scheduling for multi-objective flexible job-shop problems with flexible workdays.* Engineering Optimization, 2016. **48**(11): p. 1973-1989.
21. Gholami, M. and M. Zandieh, *Integrating simulation and genetic algorithm to schedule a dynamic flexible job shop.* Journal of Intelligent Manufacturing, 2009. **20**(4): p. 481-498.
22. Hasan, S.M.K., R. Sarker, and D. Essam, *Genetic algorithm for job-shop scheduling with machine unavailability and breakdowns.* International Journal of Production Research, 2011. **49**(16): p. 4999-5015.
23. Goren, S., I. Sabuncuoglu, and U. Koc, *Optimization of Schedule Stability and Efficiency Under Processing Time Variability and Random Machine Breakdowns in a Job Shop Environment.* Naval Research Logistics, 2012. **59**(1): p. 26-38.
24. Lei, D.-m., *Minimizing makespan for scheduling stochastic job shop with random breakdown.* Applied Mathematics and Computation, 2012. **218**(24): p. 11851-11858.
25. He, W. and D.-h. Sun, *Scheduling flexible job shop problem subject to machine breakdown with route changing and right-shift strategies.* International Journal of Advanced Manufacturing Technology, 2013. **66**(1-4): p. 501-514.
26. Mirabi, M., S.M.T.F. Ghomi, and F. Jolai, *A two-stage hybrid flowshop scheduling problem in machine breakdown condition.* Journal of Intelligent Manufacturing, 2013. **24**(1): p. 193-199.
27. Sun, D.-h., et al., *Scheduling flexible job shop problem subject to machine breakdown with game theory.* International Journal of Production Research, 2014. **52**(13): p. 3858-3876.
28. Gurel, S. and D. Cincioglu, *Rescheduling with controllable processing times for number of disrupted jobs and manufacturing cost objectives.* International Journal of Production Research, 2015. **53**(9): p. 2751-2770.
29. Nouiri, M., et al., *Two stage particle swarm optimization to solve the flexible job shop predictive scheduling problem considering possible machine breakdowns.* Computers & Industrial Engineering, 2017. **112**: p. 595-606.
30. Chen, C., Z. Ji, and Y. Wang, *NSGA-II applied to dynamic flexible job shop scheduling problems with machine breakdown.* Modern Physics Letters B, 2018. **32**(34-36).
31. Buddala, R. and S.S. Mahapatra, *Two<bold>-</bold>stage teaching-learning<bold>-</bold>based optimization method for flexible job-shop scheduling under machine breakdown.* International Journal of Advanced Manufacturing Technology, 2019. **100**(5-8): p. 1419-1432.

32. Yazdani, M., et al., *Two meta-heuristic algorithms for the dual-resource constrained flexible job-shop scheduling problem.* Scientia Iranica, 2015. **22**(3): p. 1242-1257.
33. Hansmann, R.S., T. Rieger, and U.T. Zimmermann, *Flexible job shop scheduling with blockages.* Mathematical Methods of Operations Research, 2014. **79**(2): p. 135-161.
34. Yazgan, H.R., *Selection of dispatching rules with fuzzy ANP approach.* International Journal of Advanced Manufacturing Technology, 2011. **52**(5-8): p. 651-667.
35. Aydemir, E. and H.I. Koruca, *A NEW PRODUCTION SCHEDULING MODULE USING PRIORITY-RULE BASED GENETIC ALGORITHM.* International Journal of Simulation Modelling, 2015. **14**(3): p. 450-462.
36. Bilyk, A. and L. Moench, *A variable neighborhood search approach for planning and scheduling of jobs on unrelated parallel machines.* Journal of Intelligent Manufacturing, 2012. **23**(5): p. 1621-1635.
37. Ahmadi, E., et al., *A multi objective optimization approach for flexible job shop scheduling problem under random machine breakdown by evolutionary algorithms.* Computers & Operations Research, 2016. **73**: p. 56-66.
38. Huang, S., et al., *An Improved Version of Discrete Particle Swarm Optimization for Flexible Job Shop Scheduling Problem with Fuzzy Processing Time.* Mathematical Problems in Engineering, 2016.
39. Lin, W.-C., et al., *Particle swarm optimization and opposite-based particle swarm optimization for two-agent multi-facility customer order scheduling with ready times.* Applied Soft Computing, 2017. **52**: p. 877-884.
40. Kachitvichyanukul, V., *Comparison of Three Evolutionary Algorithms: GA, PSO, and DE.* Industrial Engineeering & Management Systems An International Journal, 2012. **11**(3): p. 215-223.
41. Zhou, S., et al., *An effective discrete differential evolution algorithm for scheduling uniform parallel batch processing machines with non-identical capacities and arbitrary job sizes.* International Journal of Production Economics, 2016. **179**: p. 1-11.
42. Xu, W., Z. Ji, and Y. Wang, *A flower pollination algorithm for flexible job shop scheduling with fuzzy processing time.* Modern Physics Letters B, 2018. **32**(34-36).
43. Shambour, M.d.K.Y., A.A. Abusnaina, and A.I. Alsalibi, *Modified Global Flower Pollination Algorithm and its Application for Optimization Problems.* Interdisciplinary Sciences-Computational Life Sciences, 2019. **11**(3): p. 496-507.
44. Al-Hinai, N. and T.Y. ElMekkawy, *Robust and stable flexible job shop scheduling with random machine breakdowns using a hybrid genetic algorithm.* International Journal of Production Economics, 2011. **132**(2): p. 279-291.
45. Zhang, R., S. Song, and C. Wu, *A dispatching rule-based hybrid genetic algorithm focusing on non-delay schedules for the job shop scheduling problem.* International Journal of Advanced Manufacturing Technology, 2013. **67**(1-4): p. 5-17.
46. Kundakci, N. and O. Kulak, *Hybrid genetic algorithms for minimizing makespan in dynamic job shop scheduling problem.* Computers & Industrial Engineering, 2016. **96**: p. 31-51.
47. Gong, X., et al., *Energy- and labor-aware flexible job shop scheduling under dynamic electricity pricing: A many-objective optimization investigation.* Journal of Cleaner Production, 2019. **209**: p. 1078-1094.
48. Chan, F.T.S., T.C. Wong, and L.Y. Chan, *Flexible job-shop scheduling problem under resource constraints.* International Journal of Production Research, 2006. **44**(11): p. 2071-2089.
49. Benttaleb, M., F. Hnaien, and F. Yalaoui, *Minimising the makespan in the two-machine job shop problem under availability constraints.* International Journal of Production Research, 2019. **57**(5): p. 1427-1457.
50. Tang, H., et al., *Flexible job-shop scheduling with tolerated time interval and limited starting time interval based on hybrid discrete PSO-SA: An application from a casting workshop.* Applied Soft Computing, 2019. **78**: p. 176-194.

51. Zhang, C., Y. Rao, and P. Li, *An effective hybrid genetic algorithm for the job shop scheduling problem.* International Journal of Advanced Manufacturing Technology, 2008. **39**(9-10): p. 965-974.
52. Li, Y., X. Yao, and M. Liu, *Cloud Manufacturing Service Composition Optimization with Improved Genetic Algorithm.* Mathematical Problems in Engineering, 2019. **2019**.
53. Wu, Z. and M.X. Weng, *Multiagent scheduling method with earliness and tardiness objectives in flexible job shops.* IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society, 2005. **35**(2): p. 293-301.