An Effective Hybrid Genetic Approach for Flexible Job Shop Scheduling Problem with Machine Status Constraint

Qiaofeng Meng[1,a,*], Linxuan Zhang [1,b], Yushun Fan[1,c], Haiwei Luo[2] and Hongjie Zhao[2]

[1] National CIMS Engineering Technology Research Center at Tsinghua University, Beijing, China
[2] Capital Aerospace Machinery Company, Beijing, China
[a] wwmmyymmqqff@ 163.com, [b] lxzhang@mail.tsinghua.edu.cn, [c] fanyus@tsinghua.edu.cn

Keywords: flexible job shop scheduling problem, genetic algorithm, tabu search, local search, machine status

Abstract. In this paper, an efficient hybrid genetic algorithm (GA) algorithm is proposed to solve the flexible job shop scheduling problem (FJSSP) problem with machine status constraint. According to the machine status in the actual production workshop, the machine available constraint is added to the scheduling. The mathematical model is designed in the paper for FJSSP with the machine available constraint. The GA algorithm is applied to produce initial solution and tabu search (TS) is used to promote local search ability. The operation-based representation method, crossover and mutation operation are utilized to increase the performance. The computational experiments are conducted to testify the efficiency of the proposed algorithm.

1. Introduction

Job shop scheduling problem (JSSP) has been a hot issue in the field of industrial and combinatorial optimization. The job shop scheduling problem is defined as assigning a given set of jobs to a set of machines in the workshop to meet certain goals. Due to the high complexity, job shop scheduling problem has been proved to be a NP-hard combinatorial optimization problem[1]. However, the scheduling problem in actual shop is more flexible than the job shop scheduling problem, which has been extended to the flexible job shop scheduling problem (FJSSP). In the job shop scheduling problem, each job has a certain process route. However, for flexible job shop scheduling problem, each operation can be processed by a machine selected from a job set. Therefore, the flexible job shop scheduling problem is more complex which includes two problems: machine allocation and scheduling[2].

In recent years, many scholars have done a lot of research and made a lot of achievements in FJSSP because of the excellent scheduling system which can effectively improve the production efficiency and the competitiveness of enterprises. Because of the NP-hard property of the flexible job shop scheduling problem, it is difficult to adopt mathematical methods to obtain accurate

solution in finite time. Therefore, the meta heuristic algorithm to obtain the optimal solution through evolution naturally becomes the focus of current research.

Mohsen Ziaee [3] develop a fast heuristic algorithm based on a constructive procedure which ensure to obtain good quality schedules very quickly to the distributed and flexible job-shop scheduling problem (DFJSP) which involves the scheduling of jobs (products) in a distributed manufacturing environment under the assumption that the shop floor of each factory/cell is configured as a flexible job shop. Ling Wang et al.[4] present an effective bi-population based estimation of distribution algorithm (BEDA) which consists of two sub-populations to adjust the machine assignment and operation sequence respectively with a splitting criterion and a combination criterion. A Bagheri et al.[5] put up with an artificial immune algorithm (AIA) based on integrated approach which uses several strategies for generating the initial population and adopts different mutation operators to select the individuals for reproduction. L Wang et al.[6] design an effective artificial bee colony (ABC) algorithm for solving the flexible job shop scheduling problem with the criterion to minimize the maximum completion time (makespan) which adopts crossover and mutation operators, local search strategy based on critical path and updating mechanism of population by generating the scout bees with the initialing strategy to enhance the effectiveness of the algorithm. Y Xu et al.[7] propose a shuffled frog leaping algorithm (SFLA) based algorithm for solving the FJSP. The algorithm design a hybrid encoding scheme in which two sequences are used to illustrate both operation order sequence and machines assignment and a special bi-level crossover scheme as a local search scheme based on the critical path to promote the performance of the algorithm. They then put forward an effective teaching–learning-based optimization algorithm (TLBO) to solve the flexible job-shop problem with fuzzy processing time (FJSPF) [8]. S Wang et al. [9] present an effective estimation of distribution algorithm (EDA) to solve the flexible job-shop scheduling problem with the criterion to minimize the maximum completion time (makespan). They design a probability model to generate new individuals and utilize a local search strategy based on critical path to balance the ability of exploration and exploitation. M Akhshabi et al[10] present a clonal selection algorithm (CSA) to solve the Flexible Job-shop Scheduling Problem and the efficiency of CSA has been demonstrated by experiments. K Ida et al. [11] focus on the maximum of the workloads for all machines and propose a new method of survival selection, a method of creation of the initial solution, mutation, and a method of escape to the genetic algorithm for the FJSSP. LD Giovanni et al.[12] propose an improved genetic algorithm (IGA) to solve the distributed and flexible job-shop scheduling problem. In the algorithm, Gene encoding is extended to include information on job-to-FMU assignment, and a greedy decoding procedure exploits flexibility and determines the job routings. Furthermore, a new local search based operator is used to improve available solutions by refining the most promising individuals of each generation. M Zhang et al. [13] present an improved genetic algorithm (GA) to solve a class of complex job shop scheduling problem with characteristics of re-entrant and flexible. A comprehensive search mechanism is proposed to effectively overcome the contradiction between convergence rate and convergence accuracy of GA. Meanwhile, a partition crossover operator with competition between parents and children has also been used in GA. J Tang et al. [14] propose a novel hybrid method by combining the chaos particle swarm optimization with genetic algorithm which consists of a novel initialization method based on the improved Kacem assignments scheme and genetic operators according to the characteristic of flexible job-shop scheduling problem. Experimental results indicate that the method is efficient and competitive compared to some existing methods. A Thammano et al. [15] presents a hybrid artificial bee colony algorithm for solving the flexible job-shop scheduling problem (FJSP) with the criteria to minimize the maximum completion time (makespan). They apply several dispatching rules and the harmony search algorithm to create the initial solutions and the simulated annealing algorithm to escape from the local optimum. JQ Li et

al.[16] put up with a hybrid of particle swarm optimization (PSO) algorithm and tabu search (TS) algorithm are presented to solve the FJSP with the criterion to minimize the maximum completion time (makespan). FM Defersha et al. [17] present a mathematical model for a flexible job-shop scheduling problem incorporating sequence-dependent setup time, attached or detached setup time, machine release dates, and time lag requirements. They design a parallel genetic algorithm that runs on a parallel computing platform to the model. S Wang，J Yu [18] propose a filtered beam search (FBS) based heuristic algorithm to deal with the variant FJSP problem with maintenance activities. M. Amiri et al. [19] develop a various neighbourhood structures (VNS) related to assignment and sequencing problems to generate neighbouring solutions for FJSSP. C Lahlou [20] present a model of a practical FJSSP in which blocking constraints have to be taken into account.

From the above review, most of the studies are limited to the standard FJSSP problem, and there still exists a gap with the actual job shop scheduling. The scheduling problem of the actual production workshop is often complex, such as the machine can't be available. In order to make the research more close to the reality, some scholars added a number of factors to be considered in the actual workshop scheduling in their study, but the research results in this field are still few. In order to solve the FJSP problem with machine state constraints, this paper puts up with a mathematical model and proposes a hybrid GA and TS algorithm to the model. The algorithm uses GA for global search, and employs the TS algorithm to improve the performance of the local search.

The rest of this paper is organized as follows. The mathematical model of flexible job shop scheduling is defined in section 2. We state the hybrid GA algorithm in section 3. In section 4, the experiments are conducted to verify the effectiveness of the algorithm. In the last fifth section, we conclude the paper and propose the next research step.

2. The mathematical model

2.1. Problem Definition

There are *n* jobs to be processed on *m* machines in the shop. Each job J_i consists of a series of N_i operations $O_{ij}(i=1,2\cdots n, j=1,2,\cdots n_i)$. An operation can be processed by a machine $M_k(M_k \in M_{ij})$ from a set of machines $M_{ij}(M_{ij} \subseteq M)$, and the processing time T_{ij} of O_{ij} changes with the machine changes. FJSSP involves solving two problems: rooting and scheduling. If an operation can be processed by part of machines, the FJSSP problem is defined as a part of the flexible FJSSP problem (P-FJSSP). Otherwise, it belongs to the fully flexible FJSSP (T-FJSSP). FJSSP needs to set the start and end times for operations on machines. The objective is to complete all jobs as early as possible, which is to minimize makespan.

The constraints are as follows.

Each operation must be processed after the operation of the previous operation (if it exists) finishes;

When a job is processed on a machine it can't be interrupted;

The same job can't be processed simultaneously on two machines;

It is only when the machine is in good condition that the operation can be done on the machine.

2.2. Mathematical model

Based on the understanding of the above problems, the mathematical model of the problem is established. The symbols used in the model and the paper are defined in Table 1.

Table 1. The symbols definition

Symbols	Definition
n	The total number of the jobs
m	The total number of the machines
J_i	The *i-th* job
O_{ij}	The *j-th* operation of job J_i
N_i	Number of operations of job J_i
M_{ij}	Machine set for operation O_{ij}
Pm_{ij}	The actual available machine set for operation O_{ij}
M_k	The *k-th* machine
S_k	The status of the *k-th* machine
T_{ij}	Processing time of operating O_{ij}
Ts_{ij}	Start time of O_{ij}
C_{ij}	The completion time of O_{ij}
C_r	The rate of crossover
MUT_r	The rate of mutation

$$F = \min\left\{\max\left(C_{ij} \mid 1 \le i \le N_i, 1 \le j \le n\right)\right\} \tag{2-1}$$

s.t.

$$Ts_{ij} \ge 0 \tag{2-2}$$

$$C_{ij} = Ts_{ij} + T_{ij} \tag{2-3}$$

$$Ts_{i(j+1)} \ge C_{ij} \tag{2-4}$$

$$Pm_{ij} = \left\{M_k \mid S_k = 2, 1 \le k \le m\right\} \in \left\{M_{ij}\right\} \tag{2-5}$$

$S_k \in \{0,1,2\}$ % 0-damaged, 1-maintenance, 2-available

The above formula (2-1) defines the equation of the objective which is to minimize makespan. The formula (2-2) indicates that all jobs are processed after the start time of scheduling. The formula (2-3) defines that when a job is processed on a machine it can't be interrupted. The formula (2-4) defines that each operation must be in accordance with the sequence of processing, that is, the current operation can't be processed until previous operation (if it exists) finishes. The formula (2-5) defines that actual available machine set for operation O_{ij}. In addition, other constraints must be satisfied in the section 2.1.

3. The Proposed Hybrid Genetic Algorithm and Tabu Algorithm

There are two classes for solving FJSSPs: hierarchical approach and integrated approach. The hierarchical approach was introduced into FJSSPs by Brandimarte in 1993 [2] which attempts to solve the problem by two steps which are rooting and scheduling. The Integration method fuses the two problems together. Therefore, the integration method is more difficulties, but it produces better result.

Genetic algorithm simulates the evolutionary procedure of natural selection and natural inheritance of biology circles[21]. And it has many advantages: strong robustness, simple searching method, global parallel searching, and be suitable to solve the problems with large scale etc.. However, genetic algorithm also has its own drawbacks such as lower convergence speed, large number of repeated times, and falling into local optimal solutions easily etc. for complicated problem.

To overcome the shortcomings of GA, TS is utilized for jumping out of local optimal solution. The application of TS to the FJSSP problem has been applied by many scholars[22]. Hurink et al.[23] employ TS techniques to solve FJSSP and demonstrated its good performance.

We propose a hybrid GA and TS algorithm (NHGATS) which uses GA to search in the global space, and applies TS which has good local searching ability to exploit in the local space.

The pseudo code of the NHGATS is shown in Figure 1.

Procedure: NHGATS()
Input: The machine allocation dataset J_m of each operation,
the machining time dataset T_p of each operation,
the status dataset S_m of each operation
Output:solution s
Begin
Step 1.Initialize the parameters of the proposed NHGATS:
The population size P_s,
Generation gap G_{ap},
Crossover rate C_r,
Maximum number of iterations *maxgen*
The solution is the same for S_g generations
Step 2.Generate initial solution by GA:
Generating initial population *Chrom*;
Calculate the objective value $ObjV(s)(s \in Chrom)$
While gen<maxgen
{
FitnV=ranking(ObjV); %Assign fitness values
SelChrom=Selection(Chrom); %Perform selection operator
Crossover(SelChrom);% Perform selection operator
Mutation(SelChrom);
[Chrom ObjV]=reins(Chrom, SelChrom);
$minobjV = \min(ObjV)$;
If there is no improvement for *Sg* generations successively in *minobjV*
$T_b \leftarrow minobjV$
endif
Step 3. Local search.
EndWhile
End

Figure 1. The pseudo code of the NHGATS.

3.1. Encoding

The most popular coding methods of scheduling problems are as follows: job-based coding, operation-based coding, priority-based coding, etc. The operation-based coding is adopted in this paper. Since FJSSP contains two sub problems, rooting and scheduling. Correspondingly, the method is mainly composed of integration and hierarchical. Although the integration method is difficult, but it can take into account the relationship between machine allocations and scheduling. So the integration approach is closer to reality, and the results of it are better. In this paper, we use the integrated method, the encoding of which includes two parts: machine allocation and scheduling. For example, a flexible scheduling problem with two jobs and four machines is shown in Table 2. The first job has two operations and the second job has three operations. Machines M_1 and M_4 are available, and machines M_2, M_3 are not available.

Table 2. Machines, jobs and operations definition

Jobs	Operations	Machines			
		M_1(Available)	M_2(Damaged)	M_3(In maintenance)	M_4(available)
1	O_{11}	8	–	5	6
	O_{12}	–	9	–	7
2	O_{21}	4	6	–	–
	O_{22}	11	–	13	14
	O_{23}	–	3	2	4

From the table, the machine set of each operation is defined to be a set of available machine as follows. $M_{11}=\{M_1,M_4\}, M_{12}=\{M_4\}$,…, $M_{23}=\{M_4\}$.

The operation-based coding defines the number of jobs by an integer sequence. A job number emerged for the first time defines the first operation of the job, and the second time is defined as the second operation of the job. For example, a sequence of operations O_{21}-O_{11}-O_{22}-O_{12}-O_{23} is to be encoded as 2-1-2-1-2. The coding of the machine allocation part uses an integer to define the number of elements in the set of machine allocation. For example, for a encoding sequence of machine allocation 1-2-2-1-1, the first bit of 1 represents the first machine in the set of $M_{21}=\{M_1\}$,which is M_1 . The second bit of 2 represents the second machine in the set of $M_{11}=\{M_1,M_4\}$, which is M_4.

3.2. Selection Operation

The selection operation is to select a part of individuals in the current population into the next generation directly, and other individuals need perform crossover and mutation and then evolve to the next generation. By these operation, the elite gene is retained, and the diversity of the population is increased. For example, if the population size is S_p and the generation gap is G_{ap} , only $S_p\times(1-G_{ap})$ individuals can enter the next generation directly.

3.3. Crossover Operation

Crossover is an important operation to communicate with other individuals and enhance diversity of an individual. Each individual crosses with another one randomly selected from the population with the probability of C_r . The method described in paper[24] is adopted in this paper.

3.4. Mutation Operation

Mutation is another important operation to enhance diversity of the population. In order not to destroy the good gene, mutation is usually performed with small probability. This paper proposed two kinds of mutation operation. One is the variation of the operation on the same machine; the other is the mutation of the machine. Operation variation is carried out with probability of the mutation rate. The operation variation is defined as randomly generating two position of an individual and exchanging the two operations. The machine variation is to change the current processing machine of the operation to another one from the set of available machines of the operation.

3.5. Local Search by TS

If the current solution has been emerged for a certain times，put the current solution in the tabu list. When the current solution exists in the tabu list, the neighbourhood search is performed on the current solution. The pseudo code of the local search by TS is shown in Figure 2.

```
Procedure:LocalSearch()
Input:current solution s,tabu list T_b
Output:solution s
Begin
Step 1. Initialization
        Tabu list T_b ,
        local search length L_T ,
        Loop variable T_i
Step 2. Local search
while S ∈ T_b && T_i < L_T
  {
    S'←Neighbour(s);
    ObjV = CalObjV(s')     %Calculate the objective value
    If (ObjV<minobjV)
      minobjV = ObjV
      S←S'
    Endif
    T_i = T_i + 1
  }
EndWhile
End
```

Figure 2. The pseudo code of the local search by TS.

4. Numerical experiments

In order to achieve the best efficiency of the algorithm proposed in this paper, we have carried out experiments to select the appropriate parameters. The population size takes a value in the collection {80,100,120,160}; The set of crossover rate is {0.3, 0.5, 0.8} and the set of mutation rate is {0.2, 0.4, 0.6, 0.8}; the length of the tabu list is selected from {5, 8, 10,12}. Parameter values are selected from the parameter set one by one, and 5 independent experiments are carried out for each parameter value. The experimental results show that the optimal parameter settings are as follows. The size of population is set to 120; The crossover rate is 0.8; The mutation rate is 0.4; The length of tabu list is set to 8; The other parameters are set as follows. The Generation gap G_{ap} is set to 0.9. The local search length L_T is set to 20. The variable S_g is set to 10. The finish condition of the algorithm is set as follows: if the current solution is better than the best solution and has not improved for 20 generations, or the number of iterations is greater than 1000, then the algorithm is over.

To validate the efficiency of the proposed NHGATS algorithm, we apply it on Brandimarte's data set (BR data) [2] and compare the results with the pGA proposed by KENICHI IDA [11]. We use *Dev* as a standard for evaluating the NHGATS algorithm, as defined in formula (4-1).

$$Dev = \left[(F_{NHGATS} - F_{pGA}) / F_{pGA} \right] \times 100\% \quad (4\text{-}1)$$

Where *Dev* is the relative deviation, F_{NHGATS} is the optimal makespan obtained by our algorithm and F_{pGA} is the best makespan of pGA.

The two algorithms are run five times on the same instance. The experimental results are listed in Table 3. From the results of each experiment, the proposed NHGATS outperforms pGA for all the ten problems. In the case of MK04, MK06, MK07, MK09 and MK10, the proposed NHGATS has obvious advantages. In Table 4, the computational time of proposed NHGATS also has been compared with pGA. From the experimental results, The NHGATS algorithm costs less computational time in all ten instances than pGA obviously.

Table 3. Comparison between NHGATS and pGA in minimizing makespan.

Instances	Size	pGA	HNGATS	Dev(%)
MK01	10×6	40	40	0
MK02	10×6	26	26	0
MK03	15×8	204	204	0
MK04	15×8	63	59	-6.3
MK05	15×4	173	173	0
MK06	10×15	67	60	-10.4
MK07	20×5	141	140	-0.7
MK08	20×10	523	523	0
MK09	20×10	309	308	-0.3
MK10	20×15	230	223	-3.04

Table 4. Comparison between NHGA and other algorithms in computing time(s）.

Instances	Size	pGA	HNGATS	Dev(%)
MK01	10×6	2	1.8	-10
MK02	10×6	4	2	-50
MK03	15×8	8	4	-50
MK04	15×8	10	4.2	-58
MK05	15×4	7	5	-28.6
MK06	10×15	15	11	-26.7
MK07	20×5	17	14	-17.6
MK08	20×10	22	17	-22.7
MK09	20×10	21	16	-23.8
MK10	20×15	35	23	-34.3

5. Conclusion

In this paper, a mathematical model of flexible job shop scheduling problem with machine status constraints is proposed, and an effective hybrid genetic algorithm and TS algorithm(NHGATS) is designed to solve the model to minimize makespan. The algorithm uses GA to produce an initial solution set, and then uses TS for local search. The results show that the proposed algorithm NHGATS outperforms pGA in ten instances for the quality of solution and time consumption. The next research is to study the scheduling model and algorithm which is more in line with the actual production needs

Acknowledgements

We are grateful to the editor and anonymous reviewers for the constructive comments provided to improve the paper. The work was supported by the National Science and Technology Supporting Plan of China No. 2012BAF15G01).

References

[1] Garey M R, Johnson D S, Sethi R, The Complexity of Flowshop and Jobshop Scheduling[J]. Mathematics of Operations Research, 1(2):117-129,1976.
[2] Brandimarte P. Routing and scheduling in a flexible job shop by tabu search[J]. Annals of Operations Research, 1993, 41(3):157-183.
[3] Ziaee M. A heuristic algorithm for the distributed and flexible job-shop scheduling problem[J]. The Journal of Supercomputing, 2014, 67(1):69-83.
[4] Wang L, Wang S, Xu Y, et al. A bi-population based estimation of distribution algorithm for the flexible job-shop scheduling problem [J]. Computers & Industrial Engineering, 2012, 62(4):917-926.
[5] Bagheri A, Zandieh M, Mahdavi I, et al. An artificial immune algorithm for the flexible job-shop scheduling problem[J]. Future Generation Computer Systems, 2010, 26(4):533-541.

[6] Wang L, Zhou G, Xu Y, et al. An effective artificial bee colony algorithm for the flexible job-shop scheduling problem[J]. The International Journal of Advanced Manufacturing Technology, 2012, 60(1):303-315.
[7] Xu Y, Wang L, Wang S. An effective shuffled frog-leaping algorithm for the flexible job-shop scheduling problem[C]// IEEE, 2013:128-134.
[8] Xu Y, Wang L, Wang S Y, et al. An effective teaching–learning-based optimization algorithm for the flexible job-shop scheduling problem with fuzzy processing time[J]. Neurocomputing, 2015, 148:260–268.
[9] Wang S, Wang L, Zhou G, et al. An Estimation of Distribution Algorithm for the Flexible Job-Shop Scheduling Problem[C]// Advanced Intelligent Computing Theories and Applications. with Aspects of Artificial Intelligence -, International Conference, Icic 2011, Zhengzhou, China, August 11-14, 2011, Revised Selected Papers. DBLP, 2011:9-16.
[10] Akhshabi M, Akhshabi M, Khalatbari J. Solving flexible job-shop scheduling problem using clonal selection algorithm[J]. Indian Journal of Science & Technology, 2011, 4(10).
[11] Ida K, Oka K. Flexible job-shop scheduling problem by genetic algorithm[J]. Electrical Engineering in Japan, 2011, 177(3):28–35.
[12] Giovanni L D, Pezzella F. An Improved Genetic Algorithm for the Distributed and Flexible Job-shop Scheduling problem[J]. European Journal of Operational Research, 2010, 200(2):395-408.
[13] Zhang M, Wu K. An improved genetic algorithm for the re-entrant and flexible job-shop scheduling problem[C]// Chinese Control and Decision Conference. IEEE, 2016:3399-3404.
[14] Tang J, Zhang G, Lin B, et al. A Hybrid Algorithm for Flexible Job-Shop Scheduling Problem[J]. Procedia Engineering, 2011, 15(1):3678-3683.
[15] Thammano A, Phu-Ang A. A Hybrid Artificial Bee Colony Algorithm with Local Search for Flexible Job-shop Scheduling Problem ☆[J]. Procedia Computer Science, 2013, 20:96-101.
[16] Li J Q, Pan Q K, Xie S X, et al. A Hybrid Particle Swarm Optimization and Tabu Search Algorithm for Flexible Job-Shop Scheduling Problem[J]. 2010, 2:189-194.
[17] Defersha F M, Chen M. A parallel genetic algorithm for a flexible job-shop scheduling problem with sequence dependent setups[J]. The International Journal of Advanced Manufacturing Technology, 2010, 49(1):263-279.
[18] Wang S, Yu J. An effective heuristic for flexible job-shop scheduling problem with maintenance activities [J]. Computers & Industrial Engineering, 2010, 59(3):436-447.
[19] M. Amiri, M. Zandieh, M. Yazdani, et al. A variable neighbourhood search algorithm for the flexible job-shop scheduling problem[J]. International Journal of Production Research, 2010, 48(19):5671-5689.
[20] Lahlou C. Modelling and solving a practical flexible job-shop scheduling problem with blocking constraints[J]. International Journal of Production Research, 2011, 49(8):2169-2182.
[21] Man Z, Wei T, Xiang L, et al. Research on Multi-project Scheduling Problem Based on Hybrid Genetic Algorithm[C]// International Conference on Computer Science and Software Engineering. IEEE, 2008:390-394.
[22] Billaut J C. A tabu search algorithm for solving a multicriteria flexible job shop scheduling problem[J]. International Journal of Production Research, 2011, 49(23):6963-6980.
[23] Hurink J, Jurisch B, Thole M. Tabu search for the job-shop scheduling problem with multi-purpose machines[J]. OR Spectrum, 1994, 15(4):205-215.
[24] Meng Q, Zhang L, Fan Y. Approach of hybrid GA for multi-objective job-shop scheduling[J]. International Journal of Modeling Simulation & Scientific Computing, 2016.