Flexible Job Shop Scheduling based on Different Strategies and Algorithms

Jianjun Yu*, Shouze Li
School of Business Administration
South China University of Technology, Guangzhou
***EMAIL: yujj@scut.edu.cn**

Abstract: Flexible job shop scheduling (FJSS) is studied. Not only the operation scheduling problem is solved, but also the machine dispatch is optimized. The characteristic of FJSS is analyzed and accordingly the FJSS model is set up. The flexibility disposal is discussed and accordingly two kinds of scheduling strategies are put forward. The idea of algorithm hybrid is analyzed and accordingly immune simulated annealing algorithm is put forward. Several kinds of international FJSS Benchmark instances are simulated using different scheduling algorithms according to different scheduling strategies and their results are contrasted and analyzed. The simulation experiments indicate that the scheduling model, scheduling strategies and scheduling algorithms are fit for FJSS.

Keywords: Flexible job shop scheduling; scheduling strategy; scheduling algorithm; immune algorithm; simulated annealing

I. Introduction

In Job Shop Scheduling (JSS), each operation can be processed on only one machine which was fixed in advance. This makes the academic JSS different from the actual production, and accordingly the theory can't be applied to practice satisfactorily. Hence, it's necessary to extend JSS to Flexible Job Shop Scheduling (FJSS) in order to make the scheduling theory more practical。FJSS allow that each operation can select one from several machines. It's apparent that FJSS is more complex than JSS. Just because of FJSS's value and challenge, FJSS has attracted many scholars to do deep investigation. N. Nasr and E. A. Elsayed[1] studied the FJSS early, and constructed a typical Part Flexible Job Shop Scheduling (P-FJSS) instance. The instance has become a standard which is adopted by many scholars to validate their algorithms. CHAMBERS made careful research in FJSS with tabu search and finished a excellent doctoral dissertation[2]. Recently P. Borne, I. Kacem and S. Hammadi researched FJSS synthetically, and published a series of involved papers and some typical instances[3]. However, FJSS is a kind of combinatorial optimization problem. There are many restrictions in FJSS and it's very complicated to set up FJSS mathematics model, and it's very hard to compute it. FJSS is proved to be typical NP hard problem and has not been effectively solved. Consequently, it's necessary to put forward better scheduling strategy and scheduling method.

After deep analysis for FJSS, it can be found that in fact FJSS need to solve two aspects of problem which are resource configuration problem and operation optimization problem. The resource configuration is to distribute equipment for each operation from the optional equipment set. And the operation optimization is to schedule and arrange each operation distributed with equipment. Therefore, two different kinds of scheduling strategies which are decomposition strategy and integration strategy can be used. By decomposition strategy, the problems of equipment distribution and operation arrangement can be solved respectively; while integration strategy takes them into consideration systemically.

It always exists some defects while FJSS is solved with traditional single scheduling algorithm. The ides of algorithm hybrid has developed into an important and effective way to improve algorithm. The intention of algorithm hybrid is to combine the advantage of each single algorithm, in order to have better optimization result. In this paper, it will mix immune algorithm (IA) and simulated annealing algorithm (SA) and use two different kinds of scheduling strategies to solve FJSS.

II. FJSS Mathematical Model

FJSS is the extension of standard JSS which is more close to the practice of manufacture. There are two primary variations. One is canceling the hypothesis of standard JSS on fixing process route, namely, in FJSS each process can do select one machine from the optional machine set. the second is canceling the requirements of standard JSS on all machines need be undergone one time, namely, each job can be processed one time or several times in a machine or the job isn't processed in the machine. Others are the same as the standard JSS. So FJSS is still focus on constrained optimization, but its difficulty has been increased a lot. It is not only the optimization on process sequence, but also the optimization on resources configuration.

The FJSS mathematical model can be expressed as follows.

$$F = \min\left\{\max\left(t_{f,i,j}\mid_{1\le j\le L_i,1\le i\le N}\right)\right\} \quad (1)$$

s.t.

$$t_{s,i,j} \ge 0 \quad (2)$$

$$t_{f,i,j} = t_{s,i,j} + t_{i,j} \quad (3)$$

$$t_{s,i,j+1} \ge t_{f,i,j} \quad (4)$$

$$\sum_{m=1}^{M} x_{i,t,m} \geq 1 \ (\ t_{s,i,j} \leq t \leq t_{f,i,j}) \quad (5)$$

$$x_{i,t,a} \cdot x_{i,t,b} = 0 \ (a \neq b) \quad (6)$$

$$x_{i,t,m} \cdot x_{j,t,m} = 0 \ \ (i \neq j) \quad (7)$$

$$\sum_{m=1}^{M} y_{m,i,j} \geq 1 \quad (8)$$

$$\left| \left\{ m_m \mid y_{m,i,j} \neq 0,\ 1 \leq m \leq M,\ 1 \leq j \leq L_i \right\} \right| \leq L_i \quad (9)$$

$$\left\{ m_m \mid y_{m,i,j} \neq 0,\ 1 \leq m \leq M,\ 1 \leq j \leq L_i \right\} \in \left\{ m_1, m_2, \cdots, m_M \right\} \quad (10)$$

Here

$$x_{i,t,m} = \begin{cases} 1, & j_i \text{ processing at machine } m \text{ at time } t \\ 0, & \text{otherwise} \end{cases}$$

$$y_{m,i,j} = \begin{cases} 1, & o_{i,j} \text{ can be processed at machine } m \\ 0, & \text{otherwise} \end{cases}$$

F——objective function
N——job amount
M——machine amount
L_i——operation amount of j_i
$o_{i,j}$——j-th operation of j_i
$t_{i,j}$——process time of $O_{i,j}$
$t_{s,i,j}$——start time of $O_{i,j}$
$t_{f,i,j}$——finish time of $O_{i,j}$
$x_{i,t,m}$——decision variable on whether job j_i processed at machine m at time t
$y_{m,i,j}$——decision variable on whether operation $O_{i,j}$ processed at machine m

III. Flexible Job Shop Scheduling Strategy

According above analysis, FJSS needs to solve two problems in substance: resource allocation and process optimization. Considering FJSS consists by two closely related sub problems, this papers put forward two flexible job shop scheduling strategies: decomposition strategy and integration strategy.

Decomposition strategy

According to hierarchy relation and disassembly idea, FJSS problem is divided into two independent but related subsidiary problems. And they are solved by different algorithms. According to the internal logic between the subsidiary problems, resource configuration is the prerequisite of operation optimization, so resource configuration should be solved first, and the operation optimization should be solved later. That is to say, two algorithms are used to outer optimization and inner optimization respectively. Inner algorithm is equivalent to a subsidiary program embedded in the outer algorithm. The outer algorithm provides inner algorithm with initial basic data through machines assignments, and the inner algorithm evaluates fitness for outer algorithm though operation optimization. The basic optimization process is as follow.

Repeat
{Use outer algorithm to optimize resource configuration, create *n* machine assignment projects
Repeat
{Use inner algorithm to optimize *m* operation schedule projects in each machine assignment project}
Until inner optimization termination conditions are satisfied
Assess *n* optimization projects, obtain optimal project in the iteration}
Unitl outer optimization termination conditions are satisfied.

Integration strategy

This strategy makes FJSS approach to JSS, and use the similar solution method. Unlike the decomposition strategy that solves the problem via dividing FJSS into two independent subsidiary problems according to disassembly idea, integration strategy takes the two subsidiary problems into consideration at the same time according to the idea that all things are accomplished in one step. That is to say, solving the FJSS problem via integration strategy just need to use only one algorithm, and the optical solution will be obtained after an optimization. Obviously, in order to solve the two subsidiary problems at on step, scheduling coding and operator design are the key to integration strategy. General code for solving JSS problems is hard to express the two optimizations, so FJSS requires special coding method based on FJSS specialty, and FJSS combination characteristic, technology constraint, Lamarckian specialty of code, coding space specialty, storage brief, and decoding efficiency should be considered while coding. On the other hand, because special code is adopted, in order to ensure the legitimacy of the individual after evolvement, matching operator design is needed to make the optimization operation come true. In addition, because of the complication and specialty of FJSS, the better performance is required for the algorithm while the integration strategy is used to solve FJSS. For the sake of FJSS, a hybrid algorithm suitable for the integration strategy is put forward as follow.

IV. Immune Simulated Annealing Algorithm

Guiding search method has the nature of strong generality, in no need of specific information of the problem, which results in a waste of known information on the issue. Although the heuristic algorithm dependents greatly on the issues, it is able to construct solutions to special issues rapidly using information on the problem. So it has a better time performance. Therefore, it is desired to integrate reasonably the advantages of the two algorithms to construct

a new one. It is quite appealing to FJSS which requires both real-time performance and performance optimization. The author [6] proposed the idea of mingling immune algorithm (IA) with simulated annealing algorithm (SA), and complementing each other, leading to better optimization performance, and proved it theoretically. Based on this idea, the immune simulated annealing algorithm (ISAA) is put forward, and applied to the FJSS issue for integration strategic solution. Although recently there are other literatures [7] where mingling immune algorithm with simulated annealing algorithm, ISAA proposed by author is for FJSS problem, thus both construction ideology and algorithm flow have their own uniqueness. ISAA flow structure is shown in Fig. 1.

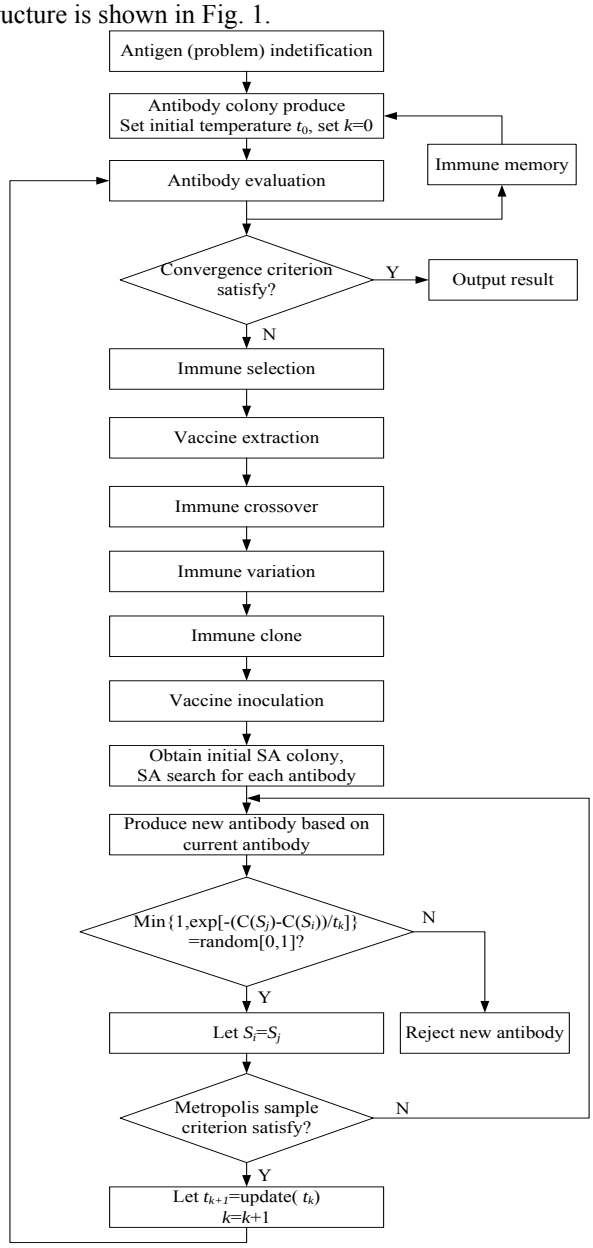

Figure.1 ISAA flow

V. Solution to FJSS

In terms of the combination features of FJSS, decomposition strategy and integration strategy were proposed previously and in this section their specific solving processes will be discussed. For the decomposition strategy, immune algorithm and simulated annealing algorithm are adopted, resource configuration through immune algorithm and operation scheduling optimization through simulated annealing algorithm. For the integration strategy, use the immune simulated annealing algorithm.

Solutions for decomposition strategy

In the decomposition strategy, two algorithms are coded and evolved independently. Due to the immune algorithm is carried out for resources configuration, the immune algorithm antibody encoding is for equipment. Each antibody is used as a machine assignment project to. The length of the antibody is equivalent to the operation amount of all jobs. It is divided by job, and each section corresponds to a job. Gene position indicates what machine is chosen during the process in sequence. Because machine amount for each operation selection is not only one, and the P-FJSS number is not in sequence and orderliness, in order to facilitate the immune operation after coding, the machine number should be transferred into another king of serial numbers by natural numbers based on some special function. And then the machine ID is used to code. This papers number arranges according to the processing time from short to long. If processing time is the same in several machines during a process, then the number arrange according the machine number from small to big. As the FJSS example in tab.1, one antibody code is expressed as Fig.2. the detailed meaning is that: operation $o_{1,1}$ is processed by machine m_5, operation $o_{1,2}$is processed by machine m_3, operation $o_{1,3}$ is processed by machine m_4, operation $o_{1,4}$ is processed by machine m_1, operation $o_{2,1}$ is processed by machine m_1, operation $o_{2,2}$is processed by machine m_4, operation $o_{3,1}$is processed by machine m_4, operation $o_{3,2}$ is processed by machine m_5, operation $o_{3,3}$ is processed by machine m_2.

Tab.1 Mapping from machine number to machine ID in *FJSS*

		Optional machine					Sequence number				
		m_1	m_2	m_3	m_4	m_5	1	2	3	4	5
J_1	$o_{1,1}$	3	7	2	6	4	m_3	m_1	m_5	m_4	m_2
	$o_{1,2}$	5	6	8	1	6	m_4	m_1	m_2	m_5	m_3
	$o_{1,3}$		3	5	2	1	m_5	m_4	m_2	m_3	
	$o_{1,4}$	2		3	3	9	m_1	m_3	m_4	m_5	
J_2	$o_{2,1}$	4		3		6	m_3	m_1	m_5		
	$o_{2,2}$				3		m_4				
J_3	$o_{3,1}$		8		3		m_4	m_2			
	$o_{3,2}$	7	4			5	m_2	m_5	m_1		
	$o_{3,3}$	6	4	7	3	1	m_4	m_5	m_2	m_1	m_3

3	5	2	1	2	1	1	2	3

Figure.2 an antibody coding example

Due to the specialty of above code, in order to avoid creating illegal antibody, every gene value should be chosen from the set of selection operation corresponding to the serial number while creating antibody. And the mutation operation also is limited in the set. Considering the inner meaning of antibody gene, usually the general two-point crossover is not used while antibody crossover, but the parallelism crossover. First, two different positive integers which are not longer than antibody lengths are created by random, and they are used as the crossover points. Then exchange crossover-section of the two parent antibodies. Then the same original genes between antibody and crossover-section are knocked out, and the missing genes are filled in according the sequence when they are in the oppose antibody. And the other one is operated like this. Antibody mutation adopts single point mutation or multi-point mutation. Mutation point amount increases with the enlargement of FJSS scale. The positive integer no more than the antibody length is created by random while mutation, and is used as mutation point. And then another serial number is chosen as new gene value from the corresponding number set by random.

After the immune algorithm evolves one generation, it will do the decoding, and machine assignment of all operations is determined and it is unique. At this time, the immune algorithm has finished its task at this phase. And next phase task will be passed to simulated annealing algorithm. Simulated annealing algorithm will start from a initial temperature, with the temperature parameters constantly decline, together with the probability jumping feature, the algorithm seeks the optimal solution by random in the solution space. In another words, it can find out the global smallest point of object function in probability purport via random search technique.

Solution of integration strategy

Integration strategy is focus on the combination of resource configuration and operation scheduling optimization, while the ISAA is also a combination of two single algorithms. So the methods is accord with the problem very much.

Coding is the first element during solving problem. Because the integration strategy optimizes two subsidiary problems at same time, two subsidiaries should be coded into a antibody while coding. Operation of job and equipment do not correspond, but processed in a machine that is selected from several machines. So the general antibody code can not be adopted. In order to make coding and decoding more convenient and efficient, the double-layer code project of job-machine is adopted. Every antibody consist two layers. The first layer is job number, and it implies the technology constraint of same job. The appearing sequence of a job stands for the operation of the job. If several operations of different jobs use the same machine, the appearing sequence in coding stands for their scheduling priority. The second layer is the number of machine the job uses. Fig.3 shows an example of a antibody. First operation of Job 2 is processed in the Machine 4. First operation of Job 1 is processed in the Machine 2. Second operation of Job 2 is processed in the Machine 9. Others are the same.

J	2	1	2	5	3	1	4	· · ·
M	4	2	9	8	3	3	8	· · ·

Figure.3 Antibody code

In order to treat the decoding more conveniently, some definitions are given as follow.

Definite 1: Machine ability space Ω : at some time, the set of the vacancy time section of the machines that can be used to process operation is called the machine ability space of the machine at this time. It can be continuous, and it also can be in section because the machine has been arranged some operations or needs repair.

Definite 2: Latest available time of machine t_w: at some time, the finish time of the last operation allocated on the machine. If no operation is arranged on the machine, then it is the scheduling starting time t_y

Definite 3: Earliest available time of machine t_z: from the time t_y, the beginning of the machine first vacancy time. If the machine has been arranged with operation in compaction, and there is no vacancy time between two processing, then it is equivalent to t_w, If no operation is arranged on the machine, it is equivalent to t_y

The antibody decoding is to search the time section earliest in Ω that can be used to process, and insert the processing time of the operation, and then alter the Ω. That is to say, the decoding mainly considers the operation as the base, and then search the processing time in the corresponding machine, namely, to find the suitable seat to be seated in Ω. First, it will judge whether the finish time of the previous operation of the same job is later than or equal to t_w. If true, the processing time of decoding operation is inserted into the Ω from the finish time of the previous operation, and the new t_w is the combination of the processing time of decoding operation and the finish time of the previous operation. Then the time section corresponding to the processing time of decoding operation is eliminated from Ω. If not, judging whether the finish time of the previous operation of the same job is earlier than or equal to t_z. If true, judges whether the vacancy time since t_z is enough for the processing time of decoding operation. If enough, then arrange the decoding operation to process since t_z. If not, then search back the earliest vacancy time that is enough to decoding operation. If the finish time of the previous operation of the same job is later than t_z, then search back the earliest vacancy time that is enough to decoding operation from the finish time of the previous operation. After the processing time inserted into the Ω, the Ω will update, and if the finish time of the decoding operation is later than original t_w, then the finish time of decoding operation is considered as new t_w. At last, after all the operations have been arranged on machine, it's necessary to adjust the operation arrangement in view of the need of optimizing the scheduling objective function, such

as improving the machine utilization ratio. Because the decoding process above imply the regulation of ahead of time as possible, on the precondition that the technology constraint is satisfied and other operations isn't affected, let the operations on each machine compress from both side to middle, and make the process more compact, and ensure less free time.

VI. The Simulation and Results Analysis

In order to verify the effectiveness of the strategies and algorithms above, and to compare their advantages and disadvantages, two kinds of benchmark instances are simulated and contrasted with different strategies and algorithms. The first kind of instance is partial flexible job shop scheduling problem with the size (job x machine) 4x6 from literature [1]. The second kind of instance is three different sizes of completely flexible job shop scheduling problems from literature [5]. They represent small scale problem, medium scale problem and large scale problem. The main parameter settings of the algorithm are shown in Tab.2. Each algorithm continuously operates 10 times for each problem.

The original data of the first kind of instance can be seen in Tab.3. The two different optimal solutions are obtained by decomposition strategy. The Gantt charts after the scheduling solutions are decoded are shown as Fig.4 and Fig.5. When use integration strategy, two different optimal solutions are also obtained. The Gantt charts after the scheduling solutions are decoded are shown as Fig.5 and Fig.6. The abscissa in the chart represents the processing time, and the ordinate is the machine number. The numbers in the box are job number and operation number. It can be found that the objective function values of the three optimal solutions are all 17, and 17 are the optimal solutions of three instances.

The specific original data of the second kind of instance can be seen in literature [5]. The results obtained by decomposition strategy and integration strategy are shown in Tab.4. The times of optimization in the table means the times of the optimal value searched during the running time. They can show the stability and the robustness of the algorithm.

Table.2 The main parameter settings of the algorithm

Q	N	N_m	P_c	P_m	P_v	k_{max}	c
4x5	30	3	0.95	0.1	0.1	50	0.8
4x6	50	5	0.95	0.1	0.1	50	0.8
10x7	80	8	0.90	0.15	0.2	100	0.90
10x10	100	10	0.85	0.2	0.2	150	0.95

Note: Q is the question type. N is colony amount. N_m is memory amount. P_c is crossover rate. P_m is mutation rate. P_v is vaccination rate. k_{max} is iteration times. c is temperature drop coefficient.

Table.3 The original data of the first kind of instance

		Optional machine					
		M_1	M_2	M_3	M_4	M_5	M_6
j_1	$o_{1,1}$	2	3	4			
	$o_{1,2}$		3		2	4	
	$o_{1,3}$	1	4	5			
	$o_{2,1}$	3		5		2	
j_2	$o_{2,2}$	4	3			6	
	$o_{2,3}$			4		7	11
	$O_{3,1}$	5	6				
j_3	$o_{3,2}$		4		3	5	
	$o_{3,3}$			13		9	12
	$o_{4,1}$	9		7	9		
j_4	$o_{4,2}$		6		4		5
	$o_{4,3}$	1		3			3

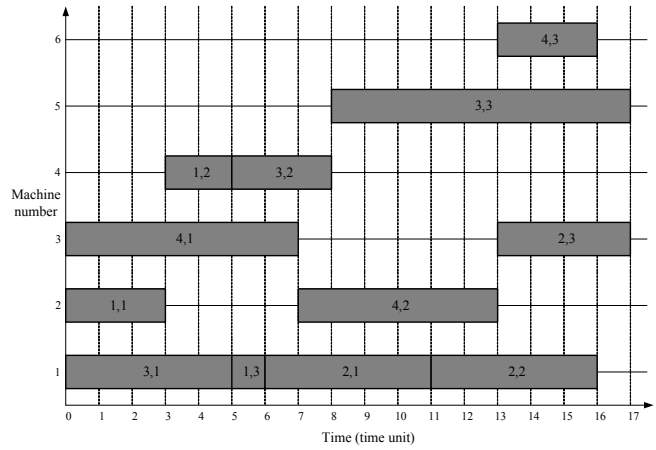

Figure.4 The Gantt chart of the first optimal solution

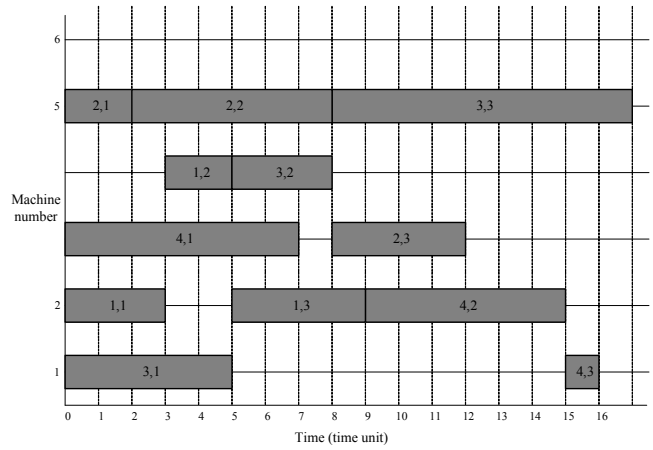

Figure.5 The Gantt chart of the second optimal solution

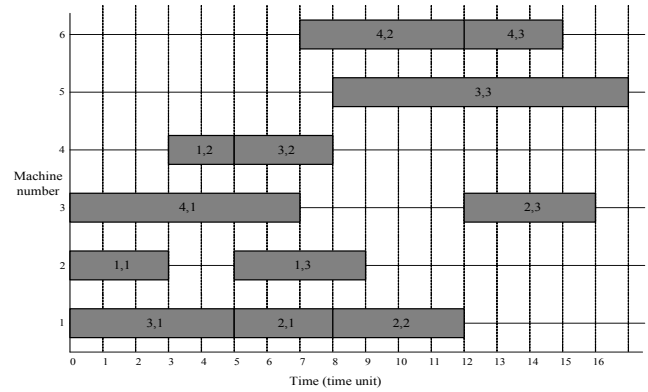

Figure.6 The Gantt chart of the third optimal solution

Table.4 Results contrast of the second kind of instance

	[5]	Decomposition			Integration		
		value	times	Time	value	times	Time
4x5	16	11	10	3	11	10	2
10x7	15	12	7	23	12	6	15
10x10	7	7	4	47	7	2	28

According to the simulation results, the optimal solutions searched are better than the optimal solutions from the original literature. The two strategies both get satisfactory results. However, comparing carefully the results solved by the two strategies, it can be found that the two kinds of results are not all the same. There are some differences between two kinds of results. The stability of the decomposition strategy is better than the integration strategy. Correspondingly, the efficiency of the integration strategy is higher than the decomposition strategy. Analyzing it deeply, it can be known that the reason for the distinct is that decomposition strategy searches the results more stably and progressively, however, integration strategy searches the results in an interactive and leaping way

VII. Conclusion

Flexible job shop scheduling is closer to the actual production than standard job shop scheduling and it is more difficult to be solved. In recent years, FJSS has always been the hot spot in research of the scheduling area. In order to make scheduling technology more perfect and more practical, this paper does some exploration work in several aspects for FJSS. The mathematics model of FJSS is analyzed, and the solving strategies of FJSS are discussed. More efficient scheduling algorithm is put forward integrating immune algorithm and simulated annealing algorithm. Several kinds of international instances are stimulated with different scheduling strategies and different algorithms, and then the stimulate results are compared and analyzed. Finally, we obtain some conclusions through exploration and research.

(1) Decomposition strategy and integration strategy are both effective to FJSS.

(2) The stability of the decomposition strategy is better than the integration strategy. Correspondingly, the efficiency of the integration strategy is higher than the decomposition strategy.

(3) ISAA integrates the advantages of IA and SA, and it is fit for the integration strategy of FJSS.

(4) IA is fit for the resource configuration and SA is fit for the operation optimization in decomposition strategy.

Acknowledgments

This project is supported by youth fund about humanities and social science of Ministry of Education (No.07JC63038), Guangdong province natural science fund (No.9451064101002828), the Fundamental Research Funds for the Central Universities, SCUT.

References

[1] N. Nasr and E. A. Elsayed. Job Shop Scheduling with Alternative Machine [J]. International Journal of Production Research.1990: 31–39.

[2] CHAMBERS, I.E. Classical and Flexible Job Shop Scheduling by Tabu Search [D]. Texas at Austin: The University of Texas, 1996: 112–119.

[3] I. Kacem, S. Hammadi, P. Borne. Approach by Localization and Genetic Manipulations Algorithm for Flexible Job-Shop Problems [J], in: Proceedings of the International IEEE Conference on Systems, Man, and Cybernetics, Tucson, AZ, USA, 2001: 2599–2604.

[4] I. Kacem, S. Hammadi, P. Borne. Direct Chromosome Representation and Advanced Genetic Operators for Flexible Job-Shop Problems [J], in: Proceedings of the International CIMCA Conference, Las Vegas, NV, USA, 2001: 123–131, 9–11.

[5] I. Kacem, S. Hammadi, P. Borne. Pareto-optimality approach for flexible job-shop scheduling problems: hybridization of evolutionary algorithms and fuzzy logic [J]. Mathematics and Computers in Simulation, 2002, 60: 245-276.

[6] Yu Jianjun, Sun Shudong, Wang Junqiang. Immune Simulated Annealing Hybrid Algorithm and Its Application for Flexible Dynamical Job Shop Scheduling [J]. China Mechanical Engineering, 2007, 18(7): 793-799.

[7] Zheng Deling, Liang Ruixin, Zhao Yuqin. A Simulated-annealing-based Immune Algotirithm for Predicting Silicon Content of Blast Furnace Hot Metal [J]. Information and Control, 2003, 32(4): 335-338.

Background of Authors

Jianjun Yu Ph.D. research interest includes production scheduling and algorithms.