Int J Adv Manuf Technol (2007) 34:783–789
DOI 10.1007/s00170-006-0652-x

ORIGINAL ARTICLE

Research on immune genetic algorithm for solving the job-shop scheduling problem

Xiao-dong Xu · Cong-xin Li

Received: 5 December 2005 / Accepted: 5 May 2006 / Published online: 20 June 2006

Abstract To solve the job-shop scheduling problem more effectively, a method based on a novel scheduling algorithm named immune genetic algorithm (IGA) was proposed. In this study, the framework of IGA was presented via combining the immune theory and the genetic algorithm. The encoding scheme based on processes and the adaptive probabilities of crossover and mutation were adopted, while a modified precedence operation crossover was also proposed to improve the performance of the crossover operator. On the other hand, the "shortest processing time" principle was selected to be the vaccine of IGA and the design method of the immune operator was given at the same time. Finally, the performance of IGA for solving JSP was validated by applying the IGA to Muth and Thompson's benchmark problems.

Keywords Immune genetic algorithm · JSSP · Vaccine

1 Introduction

The job-shop scheduling problem (JSSP) is a typical model of the manufacturing scheduling problem, which has a strong engineering background [1, 2]. JSSP belongs to the non-deterministic polynomial problem [3]. Nowadays a lot of job-shop scheduling approaches have been reported in the literature, which can be arranged under two categories: optimization and approximate/heuristic methodology [4]. Therein, approximate/heuristic algorithms mainly consist of dispatching rules, artificial intelligence, neural networks, and neighbourhood search algorithms. Furthermore, the neighbourhood search algorithms are also composed of taboo search (TS), simulated annealing (SA), ant system (AS), genetic algorithm (GA), etc., which can be named meta-heuristic algorithms. In recent years, many meta-heuristic algorithms have been studied and reported, and these can be conveniently organized as shown in Table 1.

In this study a novel scheduling algorithm named immune genetic algorithm (IGA), proposed by Jiao and Wang [12], will be applied to solve the JSSP. According to Jiao and Wang's study, IGA can improve the global exploration performance of GA by combining the immune theory in biology and the genetic algorithm, and restrain the degeneration phenomena of the colony by selectively and purposefully utilizing the characteristic information of the pending problem during the optimizing period. Up to now IGA has been popularly applied to the processes manufacturing scheduling problem [13], the travelling salesman problem (TSP) [12], and other research fields. Based on the methodology of IGA, this study aims to apply the IGA to the job-shop scheduling problem. To be exact, in this study the detail framework of IGA applied to solve JSSP will be proposed, the strategies of selecting immune vaccines and the methods of constructing the immune and other operators will be shown. Furthermore, the experiment of Muth and Thompson's basic problem [14] and the optimizing performance comparison with other scheduling algorithms will validate that applying IGA to solve JSSP is not only feasible but also effective.

X.-d. Xu (✉) · C.-x. Li
Department of Plasticity Technology,
Shanghai Jiao Tong University,
Shanghai, People's Republic of China
e-mail: xiaodong.jupiterxxu@gmail.com

2 The immune genetic algorithm [15]

The aim of leading immune concepts and methods into GA is theoretically to utilize the local characteristic information

Table 1 Meta-heuristic approaches for JSSP

Category	Detailed work
Taboo search (TS)	1. Ponnambalam et al. [5] used tabu search to solve the job shop scheduling problems with the makespan objective 2. Deng et al. [6] proposed a fast taboo search algorithm designed according to steps of the taboo search algorithm 3. Qi et al. [7] presented a fast taboo search algorithm for solving the minimum makespan problem of job shop scheduling. In the first, the insertion method is used to create the initial solution, and then the taboo search algorithm is applied to improve the final solution
Simulated annealing (SA)	1. Suresh and Mohanasundaram [8] proposed a meta-heuristic procedure based on the simulated annealing algorithm called Pareto archived simulated annealing (PASA) to discover non-dominated solution sets for the job shop scheduling problems 2. Wu et al. [9] studied a parallel simulated annealing (PSA) algorithm for the job-shop scheduling problem. In the scheduling algorithm, the whole search strategy was applied and a rule of neighbourhood search was defined to improve the performance of the individual search
Ant system (AS)	1. Wang et al. [10] proposed an improved algorithm named bi-directional convergence ant colony optimization (ACO) algorithm based on the graphic definition of the job-shop problem and the elitist strategy 2. Zhou et al. [11] proposed an ant colony algorithm (ACA for short) to solve the prematurity and unsteadiness problem in GA for JSSP with the objective of makespan minimization

for seeking the ways and means of finding the optimal solution when dealing with difficult problems. To be exact, it employs the local information to intervene in the globally parallel process and restrain or avoid the repetitive and useless work during the course, so as to overcome the blindness in action of the crossover and mutation. During the actual operation, IGA first specifically analyses the pending problem which is regarded as "antigen", where the basic characteristic information which is regarded as "vaccine" is extracted. Then the characteristic information is handled and transformed to be a solution and the set of total results from this solution is called "antibody" based on the vaccine. Finally the solution is transformed to be the immune operator by a certain format to execute the operation. To be more exact, the idea of immunity is mainly realized through two steps based on reasonably selecting vaccines, i.e. a vaccination and an immune selection, of which the former is used for raising fitness and the latter is for preventing the degeneration.

3 Design of IGA for JSSP

3.1 Mathematical model of JSSP [16]

The job-shop scheduling problem can be described by a set of n jobs with different processing sequences that is to be processed on a set of m machines. Each job has a technological sequence of machines to be processed. Each operation requires the exclusive use of each machine for an uninterrupted duration called the processing time. The time required to complete all jobs is called make-span. The objective function when solving or optimizing a JSSP problem is to determine the processing order of all operations on each machine that minimizes the make-span. A typical mathematical model of a $n/m/J/C_{\max}$ scheduling problem and the known conditions are presented as follows.

1. The set of work pieces is $P=\{ p_1, p_2, ..., p_n \}$ where p_i is the ith work piece; (i=1, 2, ..., n).
2. The set of machines is $M=\{ m_1, m_2, ..., m_m \}$ where m_j is the jth machine; (j=1, 2, ..., m).
3. The process matrix is $OP = \{op_1^T, op_2^T, \ldots, op_n^T\}$ where $op_i=\{ op_{i1}, op_{i2}, ..., op_{im} \}$ is the manufacturing process of work piece p_i. op_{ik} is the machine number of process k for work piece i. op_{ik}=0 indicates that work piece p_i is not manufactured in process k (k=1, 2, ..., m).
4. The processing time matrix is T where $T(i, j)\in T$ is the processing time of work piece p_i which is processed on machine j. When $T(i, j)=0$, it indicates that work piece p_i does not use machine j.

When solving the JSSP, the following constraints must be satisfied. The processing sequences of each work piece on a certain machine are unchangeable; every machine can only process one work piece each time; the processing time for each work piece is fixed; the start time of each machine and each work piece is zero.

According to the known conditions and constraints, the mathematical model of JSSP is described as follows.

$$min \max_{1\le k\le m} \left\{ \max_{1\le i\le n} c_{ik} \right\} \tag{1}$$

$$\begin{aligned}
s.t.\quad & c_{ik} - p_{ik} + M(1 - a_{ihk}) \ge c_{ih}, i = 1, 2, ..., n; \\
& \qquad\qquad h, k = 1, 2, ..., m \\
& c_{jk} - c_{ik} + M\left(1 - x_{ijk}\right) \ge p_{jk}, i, j = 1, 2, ..., n; k=1, 2, ..., m \\
& c_{ik} \ge 0, i = 1, 2, ..., n; k = 1, 2, ..., m \\
& x_{ijk} = 0 \ or \ 1; i, j = 1, 2, ..., n; k = 1, 2, ..., m
\end{aligned} \tag{2}$$

Equation 1 is the objective function which minimizes the maximum make-span. Equation 2 represents the process

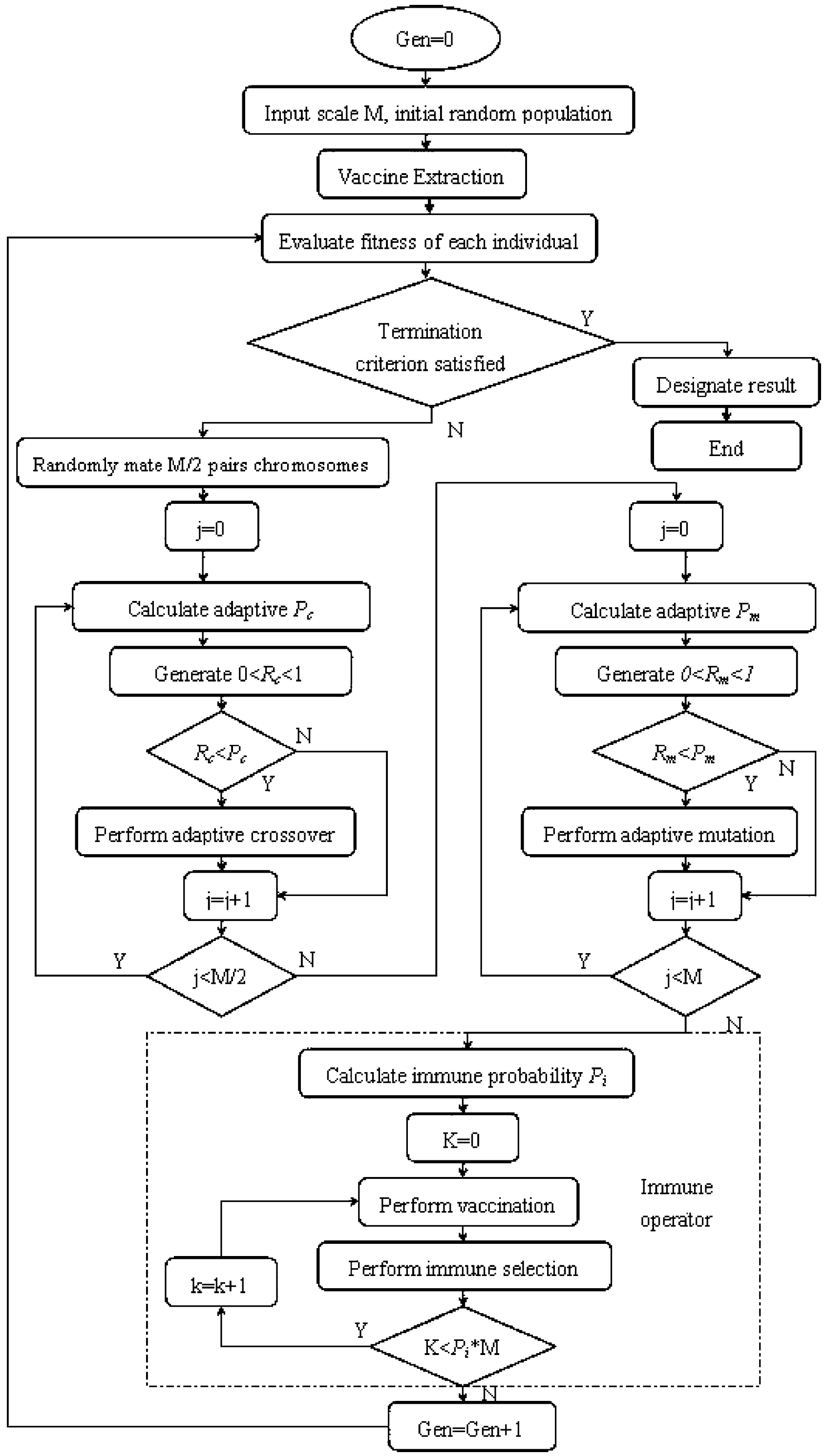

Fig. 1 Flowchart of IGA for JSSP

sequences for each work piece determined by the technology constraint and the machine sequences for each work piece which is processed on it. c_{ik}, p_{ik} denotes the end time and processing time of work piece i when processed on machine k, respectively. M denotes a positive number that is big enough. a_{ihk} and x_{ijk} denote the index coefficient and the index variable, respectively.

3.2 Framework of IGA for JSSP

In this study the immune genetic algorithm for JSSP is a certain modified genetic algorithm, namely, a simple genetic algorithm (SGA) based on the immune theory. Therefore, the basic genetic algorithm framework presented by Goldberg [17] was adopted to be the framework of IGA in this study and the primary genetic operators were modified at the same time. To be exact, we used the adaptive crossover and mutation operator instead of the original one and took advantage of the immune operator to improve the fitness of the population and avoid the degeneration of the colony through the vaccination and immune selection operation. Figure 1 shows the framework of IGA for JSSP and the detailed procedures are listed as follows.

1. Create the initial random population M, where M is the scale of the chromosome population.
2. Analyse the pending problem and collect the characteristic information to be the vaccine. In this study the vaccine is extracted according to the actual manufacturing experience of JSSP.
3. Evaluate the fitness of each individual in the population. The fitness includes maximum, minimum, average, and variance.
4. If the termination criterion is satisfied then the optimal individual is designated to be the result and the algorithm halts; or else, continues to step 5.
5. Crossover and mutation are the key operations which determine the global search ability of IGA. In this study the adaptive genetic algorithm (AGA) proposed by Srinivas and Patnaik [18] was employed. According to this methodology the probability P_c (the crossover probability or crossover rate) and the probability p_m (the mutation probability) can be automatically changed according to the fitness fluctuation of the population.
6. Do the immune operation, and then go back to step 3.

3.3 Design operators of IGA for JSSP

3.3.1 Encoding scheme of IGA for JSSP

The encoding scheme is the premier and crucial problem for IGA. In this study, an encoding method based on precedence operation proposed by Guoyong et al. [19] was adopted because this method can always gain the feasible scheduling solution after the chromosome is decoded and substituted and totally refrain from an infeasible solution. In this encoding scheme the procedure of each work piece was denoted by the corresponding work piece number and compiled according to the emerging sequence in the chromosome. For the $n/m/J/C_{\max}$ scheduling problem a certain chromosome is made up of $n \times m$ genes, the work piece number can only appear m times, and when scanning genes of this chromosome from left to right, the work piece number which appears at the kth time indicates the kth procedure of this work piece.

In this paper the infix avidity decoding scheme algorithm proposed by Chen et al. [20] was employed to be the decoding algorithm. To be exact, the chromosome was first transformed to be an ordered operation table, and then each operation was processed one by one according to the earliest processing start time based on the operation table and process constraint. According to these two steps, the scheduling solution was generated.

3.3.2 Vaccine extraction

The immune operator of IGA is composed of two operations, a vaccination and an immune selection, where the vaccine points to the basic characteristic information which is extracted from prior knowledge of the pending JSSP based on the former experience and the antibody points to a kind of solution generated from this characteristic information. The former is regarded as the estimate for the matched scheme of the best individual of pending JSSP and the latter is the sample of this scheme.

In this study the SPT (shortest processing time) dispatching rule is adopted to be the vaccine by analysing the actual manufacturing application for JSSP. When an amount of jobs is arranged on one machine, using the SPT principle, that is to say, the job with shortest processing time being arranged first, can minimize the average flowing time, delivery time, waiting time, and the average idling time. According to this characteristic of JSSP, the individual of the population is injected into the vaccine by the vaccine probability P_i. Section 3.3.5 shows the detailed steps of this operation.

3.3.3 Design of crossover operator for IGA

Crossover is the most significant operation in IGA, which determines the global search ability. In order to apply IGA to JSSP, it is important to guarantee the offspring having feasibility, variety, and a good succession for the parent generation's excellent characteristics when designing the crossover operator. Zhang et al. [21] proposed a POX (precedence operation crossover) operator based on the process encoding scheme. This operator can inherit the parent's excellent characteristics very well and make the offspring always feasible; but it did not consider whether the population can keep variety after several generations of the crossover operation. To be exact, the POX operator keeps the work piece position on the machine of the parent chromosome and makes the offspring inherit it; but it is just because of the preserved fragment of the parent chromosome that makes it possible that the individual reverted to a certain parent after several crossover operations, which leads to degeneration of the colony. Therefore, a modified POX operator is proposed in this study. This operator adds a shift step for the gene locus of the chromosome based on the original POX crossover procedure. With this step it makes the gene locus of every generation's chromosome differ from any previous generation before crossover, so it avoids the individual reverting problem in the original POX operator and takes the population variety into account at the same time. Suppose there are *Parent1* and *Parent2* in parent $n \times m$ chromosomes and they generate *Child1* and *Child2* after performing the modified POX operator. The detail procedures of the modified POX crossover operation are listed as follows.

1. Define two temp array variables *Child1*[] and *Child2*[] to store the offspring.
2. Randomly select a particular work piece number from *Parent1* and *Parent2*, then copy the genes with this work piece number to the relevant position of *Child1*[] and *Child2*[] according to their gene positions in the parent chromosome.
3. Shift the genes in *Child1*[] and *Child2*[] to the left or right according to the shift quantity that is generated randomly.
4. According to the gene sequence in *Parent1*, copy the genes with the remaining work piece numbers to the spare position of *Child2*[]; on the other hand, according to the gene sequence in *Parent2*, copy the genes with the remaining work piece numbers to the spare position of *Child1*[].

3.3.4 Design of mutation operator for IGA

When the fitness of the offspring stops evolution and does not achieve the optimal value after the crossover operation, the mutation operation can change the early mature phenomenon of IGA to a certain extent and improve the variety of the colony. In this paper a SWAP method proposed by Wang [16] was adopted to design the mutation operator. The detailed procedure is shown as follows.

1. Define one temp array variable *Child*[] to store the offspring.
2. Randomly select two different work piece numbers from *Parent*, then copy the genes with these work piece numbers to the relevant position of *Child*[] according to their gene positions in the parent chromosome.
3. Randomly generate the shift quantity and direction.
4. Shift the genes in *Child*[] according to the shift quantity and direction.
5. According to the gene sequence in *Parent*, copy the genes with the remaining work piece numbers to the spare position of *Child*[].

3.3.5 Design of immune operator for IGA

3.3.5.1 Vaccination Suppose a population is $A=\{x_1, x_2, ... x_n\}$, then the vaccination on A means the operation carried out on $n_a = P_i \times n$ individuals which are selected from A in proportion as P_i. For a certain chromosome x_i (i=1,2,...,n), the vaccination operation is to modify the genes on some particular gene positions according to the vaccine that is abstracted from the prior knowledge of the pending problem in Section 3.3.2 and make the antibody have a better fitness with bigger proportion. The detailed vaccination procedures are listed as follows.

1. For each machine M_i (i=1,2,...,m), select the work piece and make the matrix $M_{i,j} = \left[\overrightarrow{m_{0,j}}, \overrightarrow{m_{1,j}}, ..., \overrightarrow{m_{m,j}}\right]$ based on the gene sequence of this machine in the pending vaccination chromosome x_i (i=1,2,...,n), where $\overrightarrow{m_{i,j}}(i = 1, 2, ..., m; j = 1, 2, ..., m)$ is the work piece processing sequence of machine M_i.
2. For each line in matrix $M_{i,j}$, perform the sort operation according to the SPT principle.
3. According to the sorted matrix $\mathrm{M}_{i,j}$ and process constraints, revert to chromosome.

3.3.5.2 Immune selection This operation is accomplished by the following two steps. The first one is the immune test, i.e. testing the antibodies. If the fitness is smaller than that of the parent, which means serious degeneration must have happened in the process of crossover or mutation, then instead of the individual the parent will participate in the next competition; the second one is the modified preferential selection algorithm [22], whose detailed steps are shown as follows.

1. Calculate the approximate sum of fitness S_n according to Eq. 3:

   $$S_n = n \times \left(f_{avg} - f_{min}\right) \tag{3}$$

   where n is the population scale, f_{avg} is the average fitness of each generation, and f_{min} is the minimum fitness of each generation.

Table 2 Crossover and mutation probabilities for MT10×10 and MT20×5 problems

	MT10×10					MT20×5				
	GA	SyGA1	SyGA2	HTGA	IGA	GA	SyGA1	SyGA2	HTGA	IGA
p_c	0.6	0.8	0.8	0.8	–	0.6	1.0	0.6	0.8	–
p_m	0.3	0.7	0.9	0.1	–	1.0	0.8	0.7	0.1	–

GA, SyGA1, and SyGA2 are the methods proposed by Tsujimura et al. [23]

Table 3 Computational results using different methods for the MT10×10 (optimal: 930)

Method	GA	SyGA1	SyGA2	MGA	HTGA	IGA
Best	966	937	930	930	930	**930**
Average	993	965.85	965.5	953.7	943.48	**938.4**

GA, SyGA1, and SyGA2 are the methods proposed by Tsujimura et al. [23]. MGA is the method proposed by Wang and Zheng [24, 25]. HTGA is the method proposed by Liu et al. [26]

2. Calculate the individual selected proportion p_i with Eq. 4:

$$p_i = (f_i - f_{min})/S_n. \tag{4}$$

3. Sum the proportion p_i and get the total proportion g_i:

$$g_i = \sum_{j=1}^{i} p_j. \tag{5}$$

4. Randomly generate an evenly distributed number r between [0,1].
5. Compare r and g_i, if $g_{i-1} < r < g_i$, then select individual i into the next generation.

4 Computational results and comparisons

In order to test the IGA approach for JSSP in this study, the same examples as those considered by Tsujimura et al. [23], Wang and Zheng [24, 25] and Liu et al. [26], the famous MT10×10 and MT20×5 problems were simulated, and the performance of our proposed IGA approach was compared with the performances of those GA based approaches proposed in references [23–26].

The same evolutionary environments as those considered by Tsujimura et al. [23], i.e. a population scale of 40 and a maximum generation number of 5,000, were adopted in this paper. Other evolutionary environments such as the crossover probability (p_c) and the mutation probability (p_m) are summarized in Table 2. Because our IGA method employed the adaptive genetic algorithm (AGA) which calculated p_c and p_m dynamically, the p_c and p_m are denoted as "-" (means adaptive) in Table 2. Because the MGA method proposed by Wang and Zheng [24, 25] combines a GA with a simulated annealing (SA), the definition of the evolutionary environments is different from those of the GA. Therefore, the evolutionary environments in the MGA cannot be compared with those of GA-based methods.

The best makespan and the average makespan for 100 random runs using the proposed IGA approach, and those GA-based methods were, respectively, shown in Tables 3 and 4. The optimal value of each MT problem was also given in Tables 3 and 4.

According to Tables 3 and 4, it can be seen that:

1. For the MT10×10 problem, the proposed IGA approach can find the optimal makespan, the quality of which is equivalent to the MGA method given by Wang and Zheng [24, 25] and the HTGA method given by Liu et al. [26], but the average makespan obtained by using the proposed IGA approach is better than that obtained by using the other five methods.
2. For the MT20×5 problem, the proposed IGA approach gives a most optimal value, but it can achieve a smaller average value than that achieved by using the method proposed by Wang and Zheng [24, 25] and it can give a better best and average makespan than those GA-based methods given by Tsujimura et al. [23].
3. Therefore, the proposed IGA approach can be regarded as a feasible and efficient algorithm for solving the JSSP.

5 Conclusion and future work

In this paper, a novel scheduling algorithm named IGA has been applied to solve the job-shop scheduling problem. The IGA approach combines the SGA immune theory, which can improve the global exploration performance of GA. Firstly, the framework of the IGA approach was presented, and then the vaccine selection method was proposed based on the experience of the actual shop floor scheduling problem and the operators of IGA are modified. Finally, the

Table 4 Computational results using different methods for the MT20×5 (optimal: 1,165)

Method	GA	SyGA1	SyGA2	MGA	HTGA	IGA
Best	1,210	1,189	1,178	1,165	1,165	**1,168**
Average	1,251.1	1,214.9	1,233.75	1,179.22	1,172.5	**1,175.5**

GA, SyGA1, and SyGA2 are the methods proposed by Tsujimura et al. [23]. MGA is the method proposed by Wang and Zheng [24, 25]. HTGA is the method proposed by Liu et al. [26]

simulating experiments of famous MT10×10 and MT20×5 problems were executed. In these problems studied, there are numerous local optima, and therefore, these problems are challenging enough for evaluating the performances of the proposed IGA approach and those GA-based approaches proposed by Tsujimura et al. [23], Wang and Zheng [24, 25] and Liu et al. [26]. The experimental results shows that applying IGA to solve JSSP is not only feasible but also effective. But there are still issues for the author to research as future work such as the computational complexity of IGA for JSSP, or the problem of how to find a much more appropriate immune to figure out the job shop characteristics. Above all, this methodology provides a new approach to solve the job-shop scheduling problem.

References

1. Garey MR, Johnson DS, Sethi R (1976) The complexity of flowshop and job-shop scheduling. Math Oper Res 1(2):117–129
2. Jain AS, Meeran S (1999) Deterministic job-shop scheduling: past, present and future. Eur J Res 113(2):390–434
3. Lenstra JK, Rinnooy Kan AHG, Brucker P (1997) Complexity of machine scheduling problem. Ann Discr Math 1:343–362
4. Wang S-F, Zou Y-R (2003) Techniques for the job shop scheduling problem: a survey. Syst Eng Theor Pract 23(1):49–55
5. Ponnambalam SG, Aravindan P, Rajesh SV (2000) A tabu search algorithm for job shop scheduling. Int J Adv Manuf Technol 16:765–771
6. Deng Z, Huang W, Zhou L (2003) Fast taboo search algorithm for solving job shop scheduling problem. J Huazhong Univ of Sci & Tech (Nature Science Edition) 31(11):1–3
7. Qi H-Y, Huang M, Li R (2005) A fast taboo search algorithm for solving job shop problem. J DaLian Railway Institute 26(3):46–48
8. Suresh RK, Mohanasundaram KM (2005) Pareto archived simulated annealing for job shop scheduling with multiple objectives. Int J Adv Manuf Technol 29(1–2):184–196
9. Wu D-W, Lu T-D, Liu X-B, Meng Y-S (2005) Parallel simulated annealing algorithm for solving job-shop scheduling problem. Comput Integr Manuf Syst 11(6):847–850
10. Wang C-Q, Cao Y-F, Dai G-Z (2004) Bi-directional convergence ACO for job-shop scheduling. Comput Integr Manuf Syst 10 (7):820–824
11. Zhou P, Li X-P, Zhang H-F (2004) An ant colony algorithm for job shop scheduling problem. Proceedings of the 5th world congress on intelligent control and automation, Hangzhou, People's Republic of China, 15–19 June 2004, pp 2899–2903
12. Jiao L, Wang L (2000) A novel genetic algorithm based on immunity. IEEE Trans Syst Man Cybern 30(5):552–561
13. Wang J, Ma K, Feng T (2005) The application of adaptive immune genetics algorithms on hybrid-process job-shop schedule. Journal of Xi, University of Engineering Science and Technology 19(1):79–81
14. Muth JF, Thompson L (1963) Industrial scheduling. Prentice-Hall, Englewood Cliffs, NJ
15. Wang L, Jiao L (1998) The immune genetic algorithm and its convergence. Proceedings of the fourth international conference on signal processing, Beijing, China, pp 1347–1350
16. Wang L (2003) Job-shop scheduling and genetic algorithm. Tsinghua University Press, Beijing
17. Goldberg DE (1989) Genetic algorithms in search, optimization and machine learning. Addison-Wesley, New York
18. Srinivas M, Patnaik LM (1994) Adaptive probabilities of crossover and mutation in genetic algorithm. IEEE Trans Syst Man Cybern 24(4):656–667
19. Guoyong S, Hitoshi II MA, Nobuo S (1996) A new encoding scheme for job shop problems by genetic algorithm. Proceedings of the 35th conference on decision and control, Kobe, Japan, pp 4395–4400
20. Chen X, Kong Q, Wu Q (2002) Hybrid algorithm for job-shop scheduling problem. Proceeding of the 4th congress on intelligent control and automation. East China University of S&T Press, Shanghai, pp 1739–1743
21. Zhang C-Y, Rao Y-Q, Li P-G, Liu X-J (2004) An improved genetic algorithm for job-shop scheduling. Comput Integr Manuf Syst 10(8):966–970
22. Yun Q-X (2000) Evolution algorithm. Metallurgical Industry Press, Beijing
23. Tsujimura Y, Mafune Y, Gen M (2001) Effects of symbiotic evolution in genetic algorithms for job-shop scheduling. In: Proceedings of the IEEE 34th international conference on system sciences, Hawaii, pp 1–7
24. Wang L, Zheng DZ (2001) An effective hybrid optimization strategy for job-shop scheduling problems. Comput Oper Res 28:585–596
25. Wang L, Zheng DZ (2002) A modified genetic algorithm for job shop scheduling. Int J Adv Manuf Technol 20:72–76
26. Liu T-K, Tsai J-T, Chou J-H (2006) Improved genetic algorithm for the job-shop scheduling problem. Int J Adv Manuf Technol 27 (9–10):1021–1029