Solving Job-Shop Scheduling Problem Based on Improved Adaptive Particle Swarm Optimization Algorithm

Gu Wenbin(顾文斌)[1,2], *Tang Dunbing*(唐敦兵)[1*], *Zheng Kun*(郑堃)[1]

1. College of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, 210016, P. R. China; 2. College of Mechanical and Electrical Engineering, Hohai University Changzhou, Changzhou, 213022, P. R. China

(Received 4 November 2013; revised 24 March 2014; accepted 28 March 2014)

Abstract: An improved adaptive particle swarm optimization (IAPSO) algorithm is presented for solving the minimum makespan problem of job shop scheduling problem (JSP). Inspired by hormone modulation mechanism, an adaptive hormonal factor (HF), composed of an adaptive local hormonal factor (H_l) and an adaptive global hormonal factor (H_g), is devised to strengthen the information connection between particles. Using HF, each particle of the swarm can adjust its position self-adaptively to avoid premature phenomena and reach better solution. The computational results validate the effectiveness and stability of the proposed IAPSO, which can not only find optimal or close-to-optimal solutions but also obtain both better and more stability results than the existing particle swarm optimization (PSO) algorithms.

Key words: job-shop scheduling problem (JSP); hormone modulation mechanism; improved adaptive particle swarm optimization (IAPSO) algorithm; minimum makespan

CLC number: TH165 **Document code**: A **Article ID**: 1005-1120(2014)05-0559-09

1 Introduction

The job-shop scheduling problem (JSP), a typical production scheduling problem, is one of the existing combinatorial optimization problems and it has been well known as an NP-hard problem[1]. Classical JSP can be described as: There are n jobs and m machines. Each job consists of m operations, and every operation, whose processing time is already known, should be processed on different machines. At the same time, each machine can only process one job, which cannot be interrupted. Finally, the main objective of JSP is to find an optimization schedule such that makespan is minimized.

JSP can be applied to manufacture processing and effects production time and cost[2]. During the past few decades, JSP has attracted wide research attention, because it is non-deterministic polynomial-time hard (NP-hard), and it is difficult to find an exact solution in a reasonable computation time with a simple optimization algorithm. Many approximate methods, especially the evolution algorithms, have been developed to solve the problem, such as tabu search method (TA)[3], genetic algorithm (GA)[4,5], simulated annealing (SA)[6], ant colony optimization (ACO)[7] and particle swarm optimization (PSO)[2]. In this paper, we focus on exploiting an improved adaptive particle swarm optimization (IAPSO) to achieve a better solution for JSP.

Foundation items: Supported by the National Natural Science Foundation of China (51175262); the Research Fund for Doctoral Program of Higher Education of China (20093218110020); the Jiangsu Province Science Foundation for Excellent Youths (BK201210111); the Jiangsu Province Industry-Academy-Research Grant (BY201220116); the Innovative and Excellent Foundation for Doctoral Dissertation of Nanjing University of Aeronautics and Astronautics (BCXJ10-09).

* **Corresponding author**: Tang Dunbing, Professor, E-mail: d.tang@nuaa.edu.cn.

Particle swarm optimization (PSO) is a novel evolutionary algorithm inspired by the motion of a flock of birds searching for foods and was proposed by Kennedy and Eberhart[8]. As a stochastic optimization method, PSO has shown a good performance for solving combinatorial optimization problems, therefore it was used to optimize JSPs[2]. But PSO′s shortages are obvious, including the infirmness local convergence and the slow convergence speed at the last period. In order to get better solution, some improved PSO algorithms were proposed. Xia, Wu and Zhang provided a hybrid particle swarm optimization combined with simulated annealing to solve the problem of finding the minimum makespan in the job-shop scheduling environment[9]. The simulated annealing algorithm was used to avoid becoming trapped in a local optimum. By reasonably combining these two different search algorithms, an effective hybrid optimization algorithm HPSO was developed and applied to the job shop scheduling problem. Liang, et al. invented a novel PSO-based algorithm for JSPs[10]. That algorithm can effectively exploit the capability of distributed and parallel computing systems, with simulation results showing the possibility of high-quality solutions for typical benchmark problems. Sha and Hsu modified the particle position based on preference list-based representation, particle movement based on swap operator, and particle velocity based on the tabu list concept[11]. A heuristic algorithm was used to decode a particle position into a schedule. Furthermore, they applied tabu search to improve the solution quality. Fan, et al. developed an improved binary particle swarm optimization (BPSO) algorithm for solving the problem of arranging m workers to process n structures, to optimize the minimum completion time of the jobs[12]. In this improved BPSO, a new method of making initial particles was presented for searching the optimum particle in the feasible dimensional problem space. Zhang, et al. proposed a hybrid particle swarm optimization algorithm which was combined of a PSO algorithm and a tabu search (TS) algorithm to solve the multi-objective flexible job-shop scheduling problem (FJSP) with several conflicting and incommensurable objectives[13]. Lin, et al. proposed a new hybrid swarm optimization algorithm which was consisted of particle swarm optimization, simulated annealing technique and multi-type individual enhancement scheme to solve the job-shop scheduling problem[14]. He, et al. proposed a novel particle swarm optimization, which maintained the particle diversification by using the chaos mechanism, to improve the job scheduling efficiency[15]. But these algorithms above still suffer from premature convergence, and are easily trapped into local optimum. Therefore, studying new methods is still an important task to some extent.

Hormone delivery pattern to target organs is crucial to the effectiveness of their action. Hormone release can be altered by pathophysiology and differences in endocrine output mediate important intraspecies distinctions. Accordingly, the mechanisms controlling the dynamics of various hormones have lately become the object of extensive biomedical research[16]. In the medical field, Liu, Ren and Ding presented some simulation models of the hormone release. However, there are few reports on developing the optimization algorithm by using the hormone modulation mechanism[17]. In order to get a better self-adaptive and stable optimization algorithm, IAPSO, inspired from hormone modulation mechanism, is proposed and applied to JSP. The proposed IAPSO is characterized by simplifying operations, high search precision, and overcoming the problems of premature phenomenon and slow evolution.

2 JSP Model

Generally, the JSP description: Let a set of jobs $J=\{1,2,\cdots,n\}$ be processed on a set of machines $M=\{1,2,\cdots,m\}$. Job j consists of a pre-

determined sequence of task operations $\{O_{j1}, O_{j2}, O_{j3}, \cdots, O_{jm}\}$, which needs to be processed without pre-emption for a given period of time on a given machine. The processing time of every operation is fixed and known. A schedule is an assignment of operations to time slots on a machine. The makespan is the maximum completion time of the jobs and the objective of the JSP is to find a schedule that minimizes the makespan. Therefore, the ultima problem is to determine the operation sequences on the machines in order to minimize the makespan.

For the JSP conceptual model, the objective scheduling problem can be defined as follows

$$g(x) = \min\{\max_{1\leqslant k\leqslant m}\{\max_{1\leqslant i\leqslant n}\{c_{ik}\}\}\} \quad c_{ik} \geqslant 0 \tag{1}$$

$$\text{s. t.} \quad c_{ik} - t_{ik} + M(1 - a_{ihk}) \geqslant c_{ih}$$

$$a_{ihk} = 0, 1 \tag{2}$$

$$c_{jk} - c_{ik} + M(1 - x_{ihk}) \geqslant t_{jk}$$

$$x_{ijk} = 0, 1 \tag{3}$$

where c_{ik} denotes the finishing time of job $i(i=1, 2, \cdots, n)$ on machine $k(k=1, 2, \cdots, m)$, t_{ik} the processing time of job i on machine k. If job i is processed on machine $h(h=1, 2, \cdots, m)$ before processing on machine k, $a_{ihk}=1$; else $a_{ihk}=0$. If job i is processed on machine k before job $j(j=1, 2, \cdots, n)$, then $x_{ijk}=1$; else $x_{ijk}=0$. M denotes a positive number. The object function for scheduling in Eq. (1) is to minimize the maximal makespan. The restriction of job sequence is defined by Eq. (2). In Eq. (3), it indicates that every machine can process at most one job at a time.

3 Traditional PSO

Suppose that there are n particles in the L-dimensional searching space, the standard algorithm is given[18]

$$V_i(k+1) = \omega \times V_i(k) + C_1 \times rand_1() \times (X_{\text{pbest}i}(k) - X_i(k)) + C_2 \times rand_2() \times (X_{\text{gbest}i}(k) - X_i(k)) \tag{4}$$

$$X_i(k+1) = V_i(k+1) + X_i(k) \tag{5}$$

where $V(k)$ and $X(k)$ represent the velocity and the position of particle k on dimension L, $X_{\text{pbest}i}(k)$ denotes the personal best position of particle k, and $X_{\text{gbest}i}(k)$ the global best position at the present. The inertia weight w is used to coordinate the flying velocity of the particle between the global exploration and the local exploitation. The constants C_1 and C_2 are two learning factors, used to control the flying direction of the particle. $rand_1()$ and $rand_2()$ are the random variables following the uniform distribution between $[0, 1]$.

The traditional PSO design is suited to a continuous solution space, and it is difficult to address combinatorial optimization problems without modification. For better solving combinatorial optimization problems, an IAPSO, inspired by hormone modulation mechanism, is proposed to handle JSP.

4 IAPSO for JSP

Generally, the traditional PSO comprises the basic optimization concept that individuals are evolved by cooperation and competition among the individuals to accomplish a common goal. In the search process, each particle of the swarm shares the mutual information globally and benefits from the discoveries and previous experiences of all other colleagues. The hormone modulation mechanism works in biological body in the similar way. The control performance of the hormone modulation mechanism contacts each organ in the biological body and coordinates them, and the quick self-adaptation and self-organization process on the changes of inside and outside environment is realized by moderation and concurrent control among them. Inspired by hormone modulation mechanism, IAPSO is proposed to handle JSP for better searching efficiency and quality.

4.1 Hormone modulation mechanism

To describe the general hormone excreting rule $F(G)$, Farhy summarized the rule formulaically as[16]: the hormone excreting rule has characteristics with monotone and non-negative, and the hormone modulation function $F_{\text{up}}(G)/F_{\text{down}}(G)$ comply with Hill function, as follows

$$F_{\text{up}}(G)=\frac{(G/T)^n}{1+(G/T)^n} \tag{6}$$

$$F_{\text{down}}(G)=\frac{1}{1+(G/T)^n} \tag{7}$$

where $T>0$ is a threshold, $n\geqslant 1$ a Hill coefficient, and G an independent variable.

If the excreting speed S_x of hormone x is controlled by the concentration C_y of hormone y, using the Eqs. (6,7), then the term controlling the secretion of hormone x in the form is

$$S_x=aF(C_y)+S_{x0} \tag{8}$$

where a is a constant, $S_x\geqslant 0$ is independent from hormone x and controls the initial excreting speed of hormone x.

An inspiration is obtained from hormone modulation mechanism, and then an adaptive hormonal factor (HF) is designed, therefore the IAPSO is proposed.

4.2 IAPSO based on hormone regulation mechanism

4.2.1 Encoding and decoding

One of the key issues in applying PSO successfully to JSP is how to encode a schedule to a search solution, i.e. finding a suitable mapping between problem solution and PSO particle. There are two coding types of PSO for JSP in manufacturing system, which are direct encoding and indirect encodings. The direct encoding includes finishing time-based encoding, operation-based encoding, job-based encoding, random key-based encoding, and so on. The indirect encoding mainly includes machine-based encoding, priority rule-based encoding, and so on. In this article, the operation-based representation method is adopted to encode a schedule as a sequence of jobs and operations.

4.2.2 Initial population

PSO must be initialized with a starting population. Initial swarm and initial particle velocities are generated randomly. By using the swap-change method, the encoding scheme, which is mentioned above, is a handy implement to reproduce the initial population. The positions of jobs are randomly changed by the swap-change method, and the conflict problem of precedence can be avoided. There are N feasible particles which will be produced throughout the initialization procedure, where N denotes the particle swarm size, by the following algorithm:

Step 1: According to a random feasible schedule, we create an initial particle vector $\boldsymbol{O}$.

Step 2: There are two integer values p_1 and p_2 which are generated, by random. The integer values (p_1 and p_2) belong to $[1, m\times n]$, where m is the number of machine and n the number of job. Swap-change two genes in a particle vector $\boldsymbol{O}$ at the positions p_1 and p_2. For example, if $\boldsymbol{O}=$ (1 $\underline{3}$ 1 2 $\underline{2}$ 3), $p_1=2$, $p_2=5$, then $\boldsymbol{O}'=$ (1 $\underline{2}$ 1 2 $\underline{3}$ 3).

Step 3: A new particle will be produced by repeating $m\times n$ times in Step 2.

Step 4: To produce N initial feasible solutions, repeat Steps 1—3 N times.

4.2.3 Fitness function

In the particle swarm, the performance evaluation of each particle is the fitness function. Usually, fitness function can transform a set of candidate schedules into a set of positive real value. It is a feature of evaluation function that an original objective function value is mapped to a fitness value which represents relative superiority of particles. In general, we select the objective function as the fitness function. In the most job shop-scheduling problem, minimizing the maximum makespan is the most common objective function. Therefore, the reciprocal of the maximal makespan (mentioned in Eq. (1)) is selected as the fitness function, and the fitness of each particle is calculated as

$$f(x)=\frac{1}{g(x)} \tag{9}$$

The higher the fitness value is, the better the particle position in the search space is. In each generation, the particle with the highest fitness in the swarm is "gbest". The result we desired is the schedule decoded from "gbest" of the final generation.

4.2.4 Adaptive modulation factor HF

In traditional PSO, the problem solution

space is formulated as a search space, and each position in the search space is a correlated solution of the problem. The particle swarm is composed of many individual particles. The new position $X(k+1)$ of the particle i is decided by tracing the present position $X(k)$ and two "extreme points" which are the personal best solution "pbest$i$" and the global best solution "gbest$i$". To search for the optimum solution of the optimization problem, each particle should fly through the solution space, and its position will represent a potential solution. In the search process, each particle is individual, and they have little connection with the others, which causes the PSO's limitations of poor convergence and premature. In order to avoid the traditional PSO's limitations and get better solution, an adaptive HF, inspired by the hormone modulation mechanism, is devised to strengthen the information connection between each particle. By referring to Eqs. (6—8), HF is designed as follows

$$\mathrm{HF}_i = H_{\mathrm{local}} \times H_{\mathrm{global}} \tag{10}$$

where H_{local} is the adaptive local hormonal factor described in Eq. (11), and H_{global} the adaptive global hormonal factor described in Eq. (12)

$$H_{\mathrm{local}} = a\tan\left(f_i - \frac{f_{i-1}+f_{i+1}}{2}\right) + \frac{\pi}{2} \tag{11}$$

$$H_{\mathrm{global}} = a\tan\left(\frac{f_{\max}-f_{\mathrm{avg}}}{f_{\max}-f_i}\right) \tag{12}$$

where f_{i-1}, f_i and f_{i+1} represent the fitness value of the particle $i-1$, i and $i+1$, f_{avg} and $f_{\max}$ the average fitness and the maximal fitness of the individual of each generation respectively. During the evolution process of the particle swarm, HF is not fixed. If the particle's present fitness f_i is larger than the value of $(f_{i-1}+f_{i+1})/2$, it represents a high local performance of the particle i, and H_{local} will be regulated to a smaller one accordingly. In contrast, if f_i is smaller than the value of $(f_{i-1}+f_{i+1})/2$, H_{local} will be increased, which can enhance search precision. At the same time, if f_i is larger than f_{avg}, it represents a high global performance of the particle i, and the hormonal secretion (H_{global}) will be also regulated to a smaller one. But if f_i is smaller than f_{avg}, the hormonal secretion (H_{global}) will be increased, which can accelerate convergence speed. Using the adaptive HF, the proposed IAPSO can get more information among the particles and evolve the particles by cooperation and competition among the particles to accomplish an optimization goal.

Therefore, according to Eqs. (4, 5, 10), the position of particles can be updated by

$$X_i(k+1) = V_i(k+1) + X_i(k) + \mathrm{HF}_i \tag{13}$$

4.2.5 Algorithm procedure

The whole IAPSO algorithm procedure is presented below.

Step 1: Initialize parameters, including swarm size S, maximum iteration K; initialize a swarm of particles (the size of each particle is $m \times n$) with positions $X_i (1 \leqslant i \leqslant s)$ by initial particles generation mechanism; initialize velocities v_i; and $X_{\mathrm{pbest}i} = X_i$.

Step 2: Evaluate each particle's fitness by calculating the value of objective function Eq. (1) for each particle. Initialize X_{gbest} position with the particle with the lowest fitness in the swarm.

Step 3: Set iteration number k to 1.

Step 4: If k is less than or equals K, generate next swarm using Eqs. (4, 13) according to the method introduced in section 4.2.4. To update the velocities of particles, we set $w = w_{\max} - k(w_{\max} \times w_{\min})/K$ and $C_1 = C_2 = 2.0$.

Step 5: Compute each particle's fitness f_i and HF_i. Finally, update the personal best solution "pbest".

Step 6: Update the global best solution "gbest".

Step 7: Set $k = k+1$, and go to Step 4.

Step 8: Output the optimization results, and the algorithm is completed.

5 Experiments and Results

To illustrate the effectiveness and performance of IAPSO for JSP, we provide a comprehensive experimental evaluation and comparison of

the proposed IAPSO algorithm with other powerful methods. The well known standard benchmark set of Lawerence is used[19]. The chosen mothods to compare with are the traditional PSO, TSSB[20] and HGA[5]. We develop the IAPSO algorithm in ANSI C language with the environment of Microsoft Visual C++ 6.0, and simulated it with a 1.73 GHz Intel Pentium M PC. The parameters used during the experimental process in Eqs. (4,5) are defined in the following. Set $C_1 = C_2 = 2.0$, $\omega_{\max} = 1.2$, $\omega_{\min} = 0.2$, swarm size $S = 60$, maximum iteration $K = 300$.

To get the average performance of IAPSO and the traditional PSO, we run each instance twenty times and the solution quality is averaged. Regarding the performance measures, the average relative percentage deviation (ARD) of each instance is computed as follows

$$\mathrm{ARD} = \frac{\sum_{i=1}^{N} \left(\frac{f_{\mathrm{best}i} - \mathrm{BKS}}{\mathrm{BKS}} \times 100\% \right)}{N} \tag{14}$$

where N is the run times, $f_{\mathrm{best}i}$ is the makespan obtained for the ith running of an algorithm and BKS is the known minimum makespan for the problem or the best known solution for Lawerence's instances. The results of the benchmark problems for all the algorithms are provided in Table 1.

In Table 1, "Instance" means the problem name, "Size" the problem size n jobs on m machines, and the boldface represents the better solution for one instance that at least one of the four algorithms cannot obtain the best known solution, respectively. According to Table 1, IAPSO can find the best known solution with 37 instances, and the other results of LA24, LA29 and LA40 are also much better than PSO, TSSB and HGA. This fact shows that the IAPSO can obtain better solutions than the other algorithms.

The comparison of the average relative percentage deviation between the IAPSO and PSO from the thirty instances is shown in Fig. 1. From Fig. 1, the first fact shows that IAPSO can provide a better searching ability than the traditional PSO. In general, PSO is easy to be trapped in a local optimal and cannot find a better solution. From the results of Table 1 and Fig. 1, using an adaptive HF, IAPSO effectively increases the local searching ability of the original PSO for the JSP scheduling problems. Observing the curve diagram in Fig. 1, the second fact is that the difference between the IAPSO's best solution and the BKS and the IAPSO's ARD are all within 3.3%. This fact shows that the solution of IAPSO is quite stable.

Table 1 The result of benchmark problems from JSP library

Instance	Size($n \times m$)	BKS	IAPSO		PSO		TSSB	HGA
			Best solution	ARD	Best solution	ARD	Best solution	Best solution
FT06	6×6	55	55	0	55	0	55	55
FT10	10×10	930	930	0.572	1 007	8.1	930	930
FT20	20×5	1 165	1 165	0.327	1 242	6.2	1 165	1 165
LA01	10×5	666	666	0	666	0	666	666
LA02	10×5	655	655	0.244	655	7.1	655	655
LA03	10×5	597	597	0.381	597	5.3	597	597
LA04	10×5	590	590	0.537	590	2.9	590	590
LA05	10×5	593	593	0	593	0	593	593
LA06	15×5	926	926	0.453	926	3.1	926	926
LA07	15×5	890	890	0	890	0	890	890
LA08	15×5	863	863	0	863	0	863	863

Continued

Instance	Size($n\times m$)	BKS	IAPSO		PSO		TSSB	HGA
			Best solution	ARD	Best solution	ARD	Best solution	Best solution
LA09	15×5	951	951	0	951	0	951	951
LA10	15×5	958	958	0	958	0	958	958
LA11	20×5	1 222	1 222	0	1 222	0	1 222	1 222
LA12	20×5	1 039	1 039	1.299	1 039	4.6	1 039	1 039
LA13	20×5	1 150	1 150	0.941	1 150	3.6	1 150	1 150
LA14	20×5	1 292	1 292	0	1 292	0	1 292	1 292
LA15	20×5	1 207	1 207	0	1 207	0	1 207	1 207
LA16	10×10	945	945	1.248	945	14.8	945	945
LA17	10×10	784	784	0.127	784	3.1	784	784
LA18	10×10	848	848	1.005	848	8.8	848	848
LA19	10×10	842	842	0.772	842	9.5	842	842
LA20	10×10	902	902	1.136	907	6.3	902	907
LA21	15×10	1 046	1 046	3.1	1 055	30.5	1 046	1 046
LA22	15×10	927	927	1.121	935	15.5	927	935
LA23	15×10	1 032	1 032	0	1 032	0	1 032	1 032
LA24	15×10	935	936	3.3	938	30.8	938	953
LA25	15×10	977	977	2.6	983	22.9	979	986
LA26	20×10	1 218	1 218	0.640	1 218	1.7	1 218	1 218
LA27	20×10	1 235	1 235	1.187	1 252	17.1	1 235	1 256
LA28	20×10	1 216	1 216	1.225	1 216	25.7	1 216	1 232
LA29	20×10	1 152	1 165	1.642	1 179	36.8	1 168	1 196
LA30	20×10	1 355	1 355	0	1 355	0	1 355	1 355
LA31	30×10	1 784	1 784	0.631	1 784	2.3	1 784	1 784
LA32	30×10	1 850	1 850	2.399	1 850	4.6	1 850	1 850
LA33	30×10	1 719	1 719	1.141	1 719	3.6	1 719	1 719
LA34	30×10	1 721	1 721	0.513	1 721	1.2	1 721	1 721
LA35	30×10	1 888	1 888	1.113	1 888	3.1	1 888	1 888
LA36	15×15	1 268	1 268	3.1	1 291	28.8	1 268	1 279
LA37	15×15	1 397	1 397	2.8	1 442	33.2	1 407	1 408
LA38	15×15	1 196	1 196	3.2	1 228	23.1	1 196	1 219
LA39	15×15	1 233	1 233	1.8	1 233	21.6	1 233	1 246
LA40	15×15	1 222	1 224	2.3	1 236	24.3	1 229	1 241

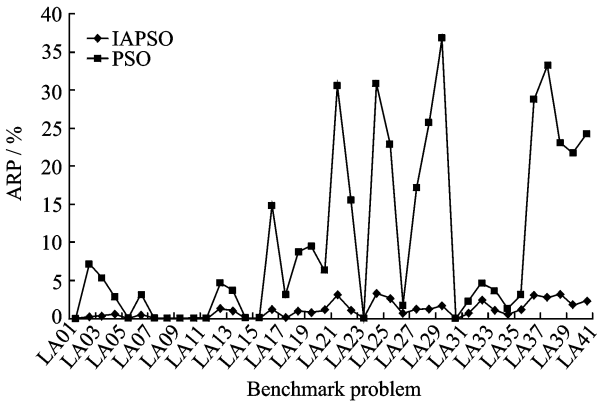

Fig. 1 Comparison of ARD between IAPSO and PSO

The average computation time on the test problems in CPU seconds is shown in Table 2. "BST" denotes the average computation time that the algorithm takes to find the best solution. Form table 2, we can see that the average "Best solution time" of the proposed IAPSO is less than the computation time of the traditional PSO. It means that the IAPSO can find the optimization schedule more efficiently than the traditional PSO.

Table 2 Computation time of the test problems

Problem	Size ($n\times m$)	PSO		IAPSO	
		BST	Total time	BST	Total time
FT06	6×6	0.0	32	0.0	29
FT10	10×10	21.7	91	5.3	157
FT20	20×5	19.2	138	18.8	210
LA01-05	10×5	5.3	50	0.5	37
LA06-10	15×5	0.1	92	0.3	65
LA11-15	20×5	0.5	143	0.1	98
LA16-20	10×10	15.5	90	13.1	137
LA21-25	15×10	37.2	164	28.9	170
LA26-30	20×10	103.1	259	93	383
LA31-35	30×10	31.4	520	3.0	751
LA36-40	15×15	68.4	254	65.2	463

Based on the above experiments, we can obtain that, by using the adaptive HF, each particle of the swarm shares the more mutual information globally, and the better solution can be got in the search space. Another advantage of the IAPSO algorithm is that it is much simpler and easy to be implemented.

6 Conclusions

IAPSO, inspired by hormone modulation mechanism, is proposed and developed for solving the JSP problem. In IAPSO, a new particles generation mechanism is presented to guarantee the particles searching optimal solution in the feasible solution space efficiently, and it also shows the strong local search ability by the adaptive HF. IAPSO is to solve a set of benchmark problems. The computational experiments show that the proposed IAPSO can find optimal or close to optimal solutions more efficiently and stably than traditional PSO. And the proposed algorithm is not worse than TSSB and HGA. Therefore, the proposed IAPSO possesses the merits of local exploration, fast convergence, and robustness to solve the JSPs.

In the future, there are three aspects for discussion as useful extension of this research: (1) How to enhance population diversity and improve search efficiency, (2) How to design more efficient information sharing mechanism and more effective local search strategy in the proposed algorithm, (3) In order to make full use of the merits in IAPSO and obtain the perfect results of JSP, the hybrid search approach should be developed, which will combine the proposed IAPSO with other search algorithms.

References:

[1] Garey M R, Johnson D S, Sethi R. The complexity of flow shop and job shop scheduling[J]. Mathematics of Operations Research, 1976, 1(2):117-129.

[2] Lian Z G, Jiao B, Gu X S. A similar particle swarm optimization algorithm for job-shop scheduling to minimize makespan[J]. Applied Mathematics and Computation, 2006,183(2):1008-1017.

[3] Nowicki E, Smutnicki C. An advanced tabu search algorithm for the job shop problem[J]. Journal of Scheduling, 2005,8(2):145-159.

[4] Leung Y, Gao Y, Xu Z B. Degree of population diversity-a perspective on premature convergence in genetic algorithms and its Markov-chain analysis[J]. IEEE Transactions on Neural Networks,1997,8(5): 1165-1176.

[5] Goncalves J F, Mendes J J M, Resende M G C. A hybrid genetic algorithm for the job shop scheduling problem[J]. European Journal of Operational Research, 2005,167(1):77-95.

[6] Suresh R K, Mohanasundaram K M. Pareto archived simulated annealing for job shop scheduling with multiple objectives[J]. The International Journal of Advanced Manufacturing Technology, 2005,29(1): 184-196.

[7] Xiang W, Lee H P. Ant colony intelligence in multiagent dynamic manufacturing scheduling[J]. Engineering Applications of Artificial Intelligence, 2008, 21(1):73-85.

[8] Kennedy J, Eberhart R C. Particle swarm optimization[C]//Proceedings of the 1995 IEEE International Conference on Neural Networks. Piscataway, NJ: IEEE Press, 1995:1942-1948.

[9] Xia W J, Wu Z M, Zhang W. Applying particle swarm optimization to job-shop scheduling problem [J]. Chinese Journal of Mechanical Engineering, 2004,17(3):437-441.

[10] Liang Y C, Ge H W, Zhou Y, et al. A particle swarm optimization-based algorithm for job-shop scheduling problems[J]. International Journal of

Computational Methods, 2005,2(3):419-430.

[11] Sha D Y, Hsu C Y. A hybrid particle swarm optimization for job shop scheduling problem[J]. Computers & Industrial Engineering, 2006,51(4):791-808.

[12] Fan K, Zhang R Q, Xia G P. Solving a class of job-shop scheduling problem based on improved BPSO algorithm [J]. Systems Engineering—Theory & Practice, 2007,27(11):111-117.

[13] Zhang G H, Shao X Y, Li P G, et al. An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem [J]. Computers & Industrial Engineering, 2009,56 (4):1309-1318.

[14] Lin T L, Horng S J, et al. An efficient job-shop scheduling algorithm based on particle swarm optimization[J]. Expert Systems with Applications, 2010, 37(3):2629-2636.

[15] He J, Jin J. Research on the job shop scheduling optimization based on CPSO algorithm[J]. Journal of Convergence Information Technology, 2012,7(11): 60-66.

[16] Farhy L S. Modelling of oscillations in endocrine networks with feedback[J]. Methods in Enzymology, 2004,384(1):54-81.

[17] Liu B, Ren L H, Ding Y S. A novel intelligent controller based on modulation of neuro-endocrine system[J]. Lecture Notes in Computer Science, 2005, 3498(3),119-124.

[18] Lei D M. A Pareto archive particle swarm optimization for multi-objective job shop scheduling [J]. Computer & Industrial Engineering, 2008,54(4): 960-971.

[19] Lawrence S. Resource constrained project scheduling: An experimental investigation of heuristic scheduling techniques[M]. GSIA. Pittsburgh, PA: Carnegie Mellon University, 1984.

[20] Pezzella F, Merelli E. A tabu search method guided by shifting bottleneck for the job shop scheduling problem[J]. European Journal of Operational Research, 2000,120(2):297-310.

(Executive editor: Zhang Bei)