Fu qing ZHAO, Jian xin TANG, Jian hua ZOU, Jonrinal DI

Lanzhou University of Technology

An Effective Hybrid Particle Swarm Optimization with Decline Disturbance Index for Expanded Job-shop Scheduling Problem

Abstract*. In this paper, an improved particle swarm optimization with decline disturbance index (DDPSO), is proposed for expanded job-shop scheduling problem (EJSSP). To balance the exploration and exploitation abilities of DDPSO, both PSO-based global search and an adaptive local search are designed and applied simultaneously in the proposed DDPSO. An index was added when the velocity of the particle is prone to stagnation in the middle and later evolution periods. The modification improves the ability of particles to explore the global and local optimization solutions, and reduces the probability of being trapped into the local optima. Theoretical analysis, which is based on stochastic processes, proves that the trajectory of particle is a Markov processes and DDPSO algorithm converges to the global optimal solution with mean square merit. After the exploration based on DDPSO, neighborhood search strategy is used in a local search and an adaptive Meta-Lamarckian strategy is employed to dynamically decide which neighborhood should be selected to stress exploitation in each generation. Simulation results and comparisons with typical algorithms show the effectiveness and robustness of the proposed DDPSO.*

Streszczenie. *Opisano ulepszony al;gorytm mrówkowy z indeksem DDPSO do rozwiązywania problemu EJSSP (expanded job-shop scheduling problem). Dodano indeks kiedy szybkość cząstki jest podatna na stagnację w środku i końcowej części procesu ewolucji. Analiza teoretyczna bazująca na ;procesach stochastycznych dowodzi że trajektoria cząstki jest procesem Markova i algorytm DDPSO pokrywa się z rozwiązaniem globalnego optimum metodą średniokwadratową. (****Efektywna hybrydowa optymalizacja algorytmu mrówkowego z indeksem DD (Decline Disturbane) do rozwiązywania problemu EJSSP (Expanded Job-shop Scheduling Problem)****).*

Keywords: Particle Swarm Optimization; Expanded Job Shop Scheduling Problem; Decline Disturbance;
Słowa kluczowe: algorytm mrówkowy (roju cząstek), EJSSP.

Introduction

The rapid changes of consumers' needs have caused a dynamic and highly volatile business environment. Manufacturing organizations are seeking efficiency gains by competing via fast time-to market and low production cost. In order to cope with and to survive in the turbulent market environment, manufacturing organizations must be capable of responding agilely to changes and adapting themselves dynamically and continuously to changes. Production scheduling plays a key role in the manufacturing systems of enterprises for maintaining a competitive position in fast-changing markets, so it is very important to develop effective and efficient advanced manufacturing and scheduling technologies and approaches [1-2]. Job shop scheduling is to schedule a set of jobs on a set of machines, which is subject to the constraint that each machine can process at most one job at a given time and the fact that each job has a specified processing order through the machines. It is not only a NP-hard problems, it also has the well-earned reputation of being one of the strong combinatorial problems in manufacturing systems. In this paper, the expanded job-shop scheduling problem (EJSSP), which is a practical production-scheduling problem with processing constraints that are more restrictive and a scheduling objective that is more general than those of the standard job-shop scheduling problem (JSSP), is considered. There are several applications of the EJSSP, especially in modern manufacturing and servicing environments. According to the research work by Yu and Liang [3], the EJSSP with more than two machines is strongly NP-hard. Thus, due to the significance both in theory and in engineering applications, it is important to develop effective and efficient novel approaches for the EJSSP.

In the previously study, JSSP has been primarily treated by mathematics methods [4-5], branch and bound methods [6] and heuristics based on priority rules [7-8]. Over the past two decades, meta-heuristics have gained wide research attention, including such topics as simulated annealing (SA) [9], tabu search [10-11], genetic algorithm (GA) [12-15], particle swarm optimization(PSO) [16-20], and scatter search(SS) [21-22].

Owing to the good performance of convergence speed in continuous and discrete problem, PSO has been adopted as a optimization methods for JSSP for decades. By modifying the particle position representation, particle movement, and particle velocity, Sha[17] constructed a particle swarm optimization (PSO) for an elaborate multi-objective job-shop scheduling problem. A General PSO (GPSO) model, which utilized crossover and mutation operations in genetic algorithm, is adopted by particles to exchange information and search randomly in candidate space. Meanwhile, in order to control the local search and ensure GPSO convergence to the global optimum solution, time-varying crossover probability and time-varying maximum step size of Tabu Search were introduced to the JSSP [23]. Optimization of the JSP, with a search space division scheme and the meta-heuristic method of PSO, by assigning each machine in a JSP with an independent swarm of particles, was obtained by multiple independent particle swarms [24]. Ivers et al. examines the optimization of the Job Shop Scheduling Problem (JSP) by a search space division scheme and use of the meta-heuristic method of Particle Swarm Optimization (PSO) to solve it [25]. Besides the velocity representation and the guide equations used by the Standard PSO, eight extensions focused on the relationship between the particles and the injection of knowledge of the problem was developed to JSP [26]. As same with many evolutional algorithms, performance of simple PSO depends on its parameters, and it often suffers the problem of being trapped in local optima so as to cause premature convergence. Many studies have been consequently carried out to prevent premature convergence and to balance the exploration and exploitation abilities [27-29]. An improved particle swarm optimization with decline disturbance index (DDPSO) is proposed to improve the ability of particles to explore the global and local optimization solutions, and to reduce the probability of being trapped into the local optima.

The remaining contents of this paper are organized as follows. In Section 2, we give the EJSSP model and the analysis for the constraints of the model. Section 3 describes the detailed DDPSO and its performance. The implementation of DDPSO for EJSSPs, by illustrating its key elements including solution representation, decoding scheme, and local search with the adaptive Meta-Lamarckian learning strategy, is presented in Section 4. In section 5, numerical test and comparisons using the popular benchmark functions are analyzed. Finally, Section 6 concludes the paper with an outlook on future work.

Expanded job-shop scheduling problrm(ejssp)h

Notations for EJSSP

The symbols for modeling scheduling problems are as follows:

n ---number of jobs;

n_i --- number of operations of job i;

m --- type number of various resources;

r_s --- number of resources of type s, $s \in [1,2,\cdots,m]$;

R_i ---set of pairs of operations $\{k,l\}$ belonging to job i, operation k precedes l;

Q_i ---set of pairs of operations $\{k,l\}$ belonging to job i;

N_q ---set of operations requiring resource q, $q \in [1,2,\cdots,r]$;

H --- large enough positive number;

t_{il} ---processing time of operation l of job i, $l \in [1,\cdots,n_i]$;

x_{ik} ---starting time of operation k of job i, $k \in [1,\cdots,n_i]$;

x_{si} ---starting time of the first (or free) operation of job i;

x_{ie} --- completion time of the last (or free) operation of job i;

a_i ---availability time of job i ;

d_i ---delivery due date of job i ;

$[i,k]$ --- the k th operation of job i, also called operation k in short if no confusion is caused

Modeling and Analysis of EJSSP

There is a predefined ordering of the tasks within a job. A machine can process only one task at a time. There are no set-up times, no release dates and no due dates. The makespan is the time from the beginning of the first task to the end of the last task, from start to finish. The aim is to find an order of the operations on each machine such that makespan is minimized.

Minimizing the total penalty for early and tardy jobs,

$$Min\,Z = \sum_{i=1}^{n}\sum_{t=1}^{d_i}\sum_{k=1}^{n_i} [z_{ik}(t)\times\max(0,d_i-x_{ie})+y_{ik}(t)\times\max(0,x_{ie}-d_i)] \tag{1}$$

A feasible solution means that the scheduling satisfies all constraint conditions. There are mainly four types of major constraints for any operation as follows:

1) Precedence constraint. Precedence constraint means that some jobs must be processed at different machines in fixed precedence sequence defined by technological planning. Concretely, the l th operation of job i must be before the k th operation of the same job, t_{ik} means processing time of operation of k of job i, if $\{k,l\} \in R_i$, i.e.

$$\sum_{i=1}^{n}\sum_{l=1}^{n_i} x_{il} - \sum_{i=1}^{n}\sum_{k=1}^{n_i} x_{ik} \ge \max\{\sum_{j=1}^{m}\sum_{t=1}^{d_i}(t+d_{ij})$$
$$y_{ij}(t), \sum_{j=1}^{m}\sum_{t=1}^{d} t y_{kj}(t) x_{il} - t_{ik}\} \tag{2}$$

2) Resource constraint. Resource constraint means that any resource can only provide service for one operation at a time; for example, resource q can only be selected by one job waiting to be processed in the queue at any time.

$$\sum_{i=1}^{n}\sum_{j=1}^{n_i} x_{ij} - \sum_{j=1}^{ni}\sum_{k=1}^{m} x_{jk} + \prod_{i=1}^{n}\sum_{j=1}^{m_i}\sum_{t=1}^{d} y_{ij}(t)(x_{ij}-x_{jk}) +$$
$$\sum_{k=1}^{n}\sum_{l=1}^{n_i} H(1-z_{kl}) \ge 0 \;\; if\,\{k,l\} \in N_q \tag{3}$$

The improved PSO algorithm with a decline disturbance index(DDPSO)

In this paper, an improved PSO algorithm, based on standard PSO algorithm, improves the velocity updating formula by adding a decline disturbance index. The modified algorithm can be described as follows:

$$\begin{cases} v_{id}^{t+1} = \omega v_{id}^{t} + c_1 r_1 (p_{id} - x_{id}^{t}) + c_2 r_2 (p_{gd} - x_{id}^{t}) + l r_3 \\ x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \end{cases} \tag{4}$$

where $l = -d_1(x - d_2)$, which is a linear decline function controlled by parameters d_1 and d_2. The variable r_3 is a random positive number, drawning from uniform distribution [0,1]. In addition, $x = t\Delta x$, both d_1 and d_2 are small constant parameters that can be set dynamically. t is the tth iteration index that has been carried out and Δx is a interval whose length can be adjusted according to the objective functions. During the evolution process of the algorithm, the disturbance index declines at a certain rate, and has minimal impact on the evolution of the particles at last, thus making the algorithm possible to converge to global optimization solution.

Early in the evolution process of PSO algorithm, for the reasons that the particles have high velocity to converge to near optima, and the algorithm has strong ability to explore global optimization solution, therefore, the impact of the decline disturbance index on it can be ignored. With the increase in the number of iterations, the velocity of the particles in the latter periods of the gradual evolution is prone to stagnation or relatively unchanged due to the convergence of the particles. The decline disturbance index of velocity update equation (8) is going to maintain the tendency of local search capability at this point, which improves the performance of the particles to jump out of local minimum solutions, and helps the particles to avoid the possibility of being trapped into local optimization solution. The mathematical model of the dynamic evolutionary equations can be simply described as follows:

$$\begin{cases} v(t+1) = \omega v(t) + c_1 r_1 (p_i(t) - x(t)) + c_2 r_2 (p_g(t) - x(t)) + l r_3 \\ x(t+1) = x(t) + v(t+1) \end{cases} \tag{5}$$

According to the assumption of the current literature [30], ω, $p_i(t)$ and $p_g(t)$ are time-invariants. Define $\varphi_1 = c_1 r_1$, $\varphi_2 = c_2 r_2$ and $A = l r_3$, so equation (5) can be shortened as:

$$\begin{cases} v(t+1) = \omega v(t) + \varphi(p - x(t)) + A \\ x(t+1) = x(t) + v(t+1) \end{cases} \tag{6}$$

where $\varphi = \varphi_1 + \varphi_2$, ω and p are constants.

Let $y(t) = p - x(t)$, so equation(6) can be written as:

$$\begin{cases} v(t+1) = \omega v(t) + \varphi y(t) + A \\ y(t+1) = -\omega v(t) + (1-\varphi) y(t) \end{cases} \tag{7}$$

and the initial condition is as follows:

(8) $$\begin{cases} v(1) = \omega v(0) + \varphi y(0) + A \\ y(1) = -\omega v(0) + (1-\varphi) y(0) \end{cases}$$

and the matrix form of equation(8) is

(9) $$\begin{bmatrix} v_{t+1} \\ y_{t+1} \end{bmatrix} = \begin{bmatrix} \omega & \varphi \\ -\omega & 1-\varphi \end{bmatrix} \begin{bmatrix} v_t \\ y_t \end{bmatrix} + \begin{bmatrix} A \\ 0 \end{bmatrix}$$

DDPSO for EJSSP

Owing to the continuous nature of DDPSO, it cannot be directly applied to EJSSPs. The key issue of applying DDPSO to EJSSPs is to find a suitable mapping between job sequence and individuals (continuous vectors) in DDPSO. For the n -job and m -machine problem, each vector contains $n \times m$ number of dimensions corresponding to $n \times m$ operations.

In this section, we will explain the implementation of DDPSO for EJSSPs by illustrating its key elements including solution representation, decoding scheme, and local search with the adaptive Meta-Lamarckian learning strategy.

Solution Representation

In this paper, we present a smallest-order-value (SOV) rule based on random key representation to convert individuals in DDPSO $X_i = [x_{i,1}, x_{i,2}, \cdots x_{i,n \times m}]$ to the job solution $S_i = [s_{i,1}, s_{i,2}, \cdots s_{i,n\times m}]$.

In the SOV rule, $X_i = [x_{i,1}, x_{i,2}, \cdots x_{i,n \times m}]$ are firstly ranked by random key representation to get a permutation of jobs $Pi = [p_{i,1}, p_{i,2}, \cdots p_{i,n\times m}]$. Then the job solution s_i is calculated by the following formula:

$$si,k = \left\lceil \frac{p_{i,k}-1}{m} \right\rceil + 1$$

Where $\lceil t \rceil$ is an operator to get the integral part of t.

The Decoding Scheme

Due to the constraints of EJSSPs, it is a primary issue to decode a solution. As we know, the set of active schedules is a subset of the semi-active ones and an optimal solution must be an active schedule to regular index such as the makespan $C\max$ and maximum tardiness $T\max$. By converting the semi-active schedule to the active one, we can improve the solution to minimize two objectives, that is, makespan and maximum tardiness. So, when decoding we check the possible idle interval before appending an operation at the last position, and fill the first idle interval before the last operation (called active decoding).For a n job and m -machine EJSSP, the machine sequence matrix M and processing time matrix T are given, where $M_{i,j}$ represents the number of the machine processing the j th operation of job i and $T_{i,j}$ represents the processing time for job j on machine i. The algorithm to decode a job solution s_i into an active schedule is described as follows:

Step 1: Set $k(w) = 1, w = 1.\cdots, n, \quad j = 1$.

Step 2: Determine the number of the machine processing the job $s_{i,j}$ denoted by $M_{s_{i,j},k(s_{i,j})}$.

Step 3: Update $k(s_{i,j}) = k(s_{i,j}) + 1$.

Step 4: Draw the job $s_{i,j}$ on the Gantt chart following the machine $M_{s_{i,j},k(s_{i,j})}$, which the earliest possible starting time and the processing time interval, where the earliest possible starting time is obtained by keeping the ending time of each processed operation and not allowing the next operation of the same job to start earlier than that time.

Step 5: Set $j = j+1$. If $j \le n \times m$ then go to step 2.

Adaptive Meta-Lamarckian Learning Strategy

Inspired by the idea of Ong and Keane's work[31], we not only apply multiple neighborhoods, but we also adopt an adaptive meta-Lamarckian learning strategy to decide which neighborhood is to be chosen for the local search so as to reward the utilization times of those neighborhoods resulting in solution improvement. The adaptive meta-Lamarckian learning strategy in our DDPSO is illustrated as follows.

Step 0: p _insert denotes the select probability of insert, p _inverse denotes the select probability of inverse, p _interchange denotes the select probability of interchange. n _insert, n _inverse and n _interchange are used to dynamically record the performance of each corresponding neighborhood.

All these variables are global variables in DDPSO.

Step 1: Select an individual $X_i(0)$ in the population and convert it to a job solution s_i according to the SOV rule.

Step 2: Set $j = 1$ n _insert= 0, n _inverse=0 and n _interchange=0.

Step 4: Randomly select u and v, where $u \ne v$, and set $s_{i\_new} = inverse(s_i, u, v)$. If s_{i_new} dominates s_i then set n _inverse= n _inverse+1 and $s_i = s_{i\_new}$.

Step 5: Randomly select u and v, where $u \ne v$, and set $s_{i\_new} = \text{int}erchange(s_i, u, v)$. If s_{i_new} dominates s_i then set n _interchange= n _interchange+1 and $s_i = s_{i\_new}$.

Step 6: Set $j = j+1$. If $j \le (n\times m)\times(n\times m+1)$ then go to step 3.

Step 7: Calculate the select probability of each neighborhood.

Set:
p _insert= n _insert/(n _insert+ n _inverse+ n _interchange), p _inverse= n _inverse/(n _insert+ n _inverse+ n _interchange), p _interchange= n _interchange/(n _insert+ n _inverse+ n _interchange).

In addition, we divide the local search into learning phase and working phase. During the learning phase, the local search with each different neighborhood is employed for the same steps, i.e., $(n\times m)\times(n\times m-1)$ steps.

Simulation Numerical test and comparisons

Experimental Setup

To test the performance of the proposed DDPSO, a computational simulation is carried out with some well studied benchmarks. For the problems used as test case, two regular criteria, the makespan $C\max(s)$ and the maximum tardiness $T\max(s)$, are used as criteria. We set the two objectives as follows (s is the job solution):

$$\text{Minimize } f_1(s) = C\max(s)$$
$$f2(s) = T\max(s)$$

The performances used in the parameters tuning are "Best", "Mean", "Minimum", and "Best rate", where the "Best", "Mean" and "Minimum" stand for the best one, the mean one and the minimum aspects of the objective values achieved in 100 runs. The "Best rate" is the rate to reach the best value. The algorithm runs 100 times with different random seeds for each parameter setting to test the random effect on the solution.
Therefore, the parameters with highest "Best rate" are better than others.

From the results shown in Table 1, we can see that all the test cases obtained the best value. It is proved that DDPSO has the good performance in searching the best value. Meanwhile, the mean and worst value we obtained are much closed to the best value ever be reported in literatures. Therefore, DDPSO is robust to EJSSP.

Results Comparison between DDPSO and Other PSO based Algorithm

In order to testify the effective and efficiency of DDPSO, the results reported in Xia and Wu [31], Sha and Hsu [16] were compared with the results obtained under DDPSO, as shown in Table 2.

Table 1. Comparison results between DDPSO and other PSO based algorithm

Test cases	$n \times m$	C^*	DDPSO	HPSO[31]	PSO-Priority[16]	PSO-Per[17]	HPSO[16]
FT06	6×6	55	55	55	55	55	55
FT10	10×10	930	930	930	1007	937	930
FT20	20×5	1165	1165	1178	1242	1165	1165
LA01	10×5	666	666	666	681	666	666
LA06	15×5	926	926	926	926	926	926
LA11	20×5	1222	1222	1222	1222	1222	1222
LA16	10×10	945	945	945	1006	945	945
LA21	15×10	1046	1046	1047	1201	1055	1046
LA26	20×10	1218	1218	1218	1409	1218	1218
LA31	30×10	1784	1784	1784	1886	1784	1784
LA36	15×15	1268	1268	1269	1437	1291	1268

Table 2. Results by using DDPSO

Test cases	$n \times m$	C^*	Best	Mean	Minimum	CPU time
FT06	6×6	55	55	55	55	6.2
FT10	10×10	930	930	944.5	956	152.2
FT20	20×5	1165	1165	1182.5	1201	291.3
LA01	10×5	666	666	666	666	14.5
LA06	15×5	926	926	926	926	50.9
LA11	20×5	1222	1222	1222	1222	129.4
LA16	10×10	945	945	945.5	946	133.3
LA21	15×10	1046	1046	1052.5	1059	724.5
LA26	20×10	1218	1218	1282.4	1218	2389.6
LA31	30×10	1784	1784	1784	1784	3698.2
LA36	15×15	1268	1268	1269	1290	3356.2

From Table 2, we can see that for all benchmark problems we selected, DDPSO has the best performance compared with that of obtaining from other 4 algorithms. Therefore, the neighbor searching method and learning strategy are more effective than that of used by other PSO based algorithms.

Conclusions

Dynamic rescheduling method is widely used in modern production plant. In this paper, the EJSSP is solved by an improved particle swarm optimization with decline disturbance index (DDPSO), which has an advantageous over standard PSO on the convergence speed and convergence feature. By simulation in the actual production workshop, the model and algorithm were testified as effective to the expanded job shop-scheduling problem in manufacturing system. It is worth pointing out that the test cases studied in this work is not very large. We will explore the efficiency of our model and approach on those problems with large number of decision variables in the future. The future work also includes the studies on how to extend our model and algorithm to solve constrained or discrete multi-objective optimization problems.

Acknowledgment

This work is financially supported by the National Natural Science Foundation of China under Grant No.61064011. And it was also supported by China Postdoctoral Science Foundation, Science Foundation for The Excellent Youth Scholars of Lanzhou University of Technology, and Educational Commission of Gansu Province of China under Grant No.20100470088, 1014ZCX017 and 1014ZTC090, respectively.

REFERENCES

[1] J. Q. Fu, C. M. Ye, J. H. Xie, etc., "Research of improved hybrid quantum algorithm in job shop scheduling problems", Computer Engineering and Applications, vol.45, No.30, pp48-52, October 2009.
[2] B. Cai, F. Mao, L. Fu, etc., "Hybrid algorithm of particle swarm optimization and stimulated annealing for job-shop scheduling", Application Research of Computers, vol.27, No.3, pp856-859, 2010.
[3] H. B. Yu, L. Wei, "Neural network and genetic algorithm-based hybrid approach to expanded job-shop scheduling", Computers & Industrial Engineering, vol.39, pp337-356, 2001.
[4] G. Chryssolouris, V. Subramaniam, "Dynamic scheduling of manufacturing job shops using extreme value theory", Production Planning & Control, vol.11, No.2, pp122-132, 2000.

[5] J. S. Smith, B. A. Peters, A. Srinivasan, "Job shop scheduling considering material handling", International Journal of Production Research, vol.37, No.7, pp1541-1560, 1999.
[6] N. Zribi, I. Kacem, A. El Kamel, etc., "Assignment and scheduling in flexible job-shops by hierarchical optimization", IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews, vol.37, pp652-661, 2007.
[7] F. Ghedjati, "Heuristics and a hybrid meta-heuristic for a generalized job-shop scheduling problem", IEEE Congress on Evolutionary Computation, pp1-8, July 2010.
[8] I. Gonzalez-Rodriguez, J. J. Palacios, C. R. Vela, etc., "Heuristic local search for fuzzy open shop scheduling", 2010 IEEE International Conference on Fuzzy Systems, pp1-8, 2010.
[9] R. Zhang, C. Wu, "A simulated annealing algorithm based on block properties for the job shop scheduling problem with total weighted tardiness objective", vol.38, No.5, pp854-867, 2011.
[10] C. Y. Zhang, P. G. Li, Z. L. Guan, etc., "A tabu search algorithm with a new neighborhood structure for the job shop scheduling problem", Computers and Operations Research, vol.34, pp3229-3242, 2007.
[11] A. baykasoglu, L. Ozbakir, A. I. Sonmez, "Using multiple objective tabu search and grammars to model and solve multi-objective flexible job shop scheduling problems", Journal of Intelligent Manufacturing, vol.15, No.6, pp777-785, 2004.
[12] M. Sakawa, R. Kubota, "Two-objective fuzzy job shop scheduling through genetic algorithm", Electronics and Communications in Japan Part Iii-Fundamental Electronic Science, vol.84, No.4, pp60-68, 2001.
[13] L. De Giovanni, F. Pezzella, "An Improved Genetic Algorithm for the Distributed and Flexible Job-shop Scheduling problem", European Journal of Operational Research, vol.200, No.2, pp395-408, 2010.
[14] J. A. Vazquez-Rodriguez, S. Petrovic, "A new dispatching rule based genetic algorithm for the multi-objective job shop problem", Journal of Heuristics, vol.16, No.6, pp771-793, 2010.
[15] C. Gutierrez, I. Garcia-Magarino, "Modular design of a hybrid genetic algorithm for a flexible job-shop scheduling problem", Knowledge-Based Systems, vol.24, No.1, pp102-112, 2011.
[16] D. Y. Sha, C. Y. Hsu, "A hybrid particle swarm optimization for job shop scheduling problem", Computers and Industrial Engineering, vol.51, pp791-808, 2006.
[17] D. Y. Sha, L. Hsing-Hung, "A multi-objective PSO for job-shop scheduling problems", Expert Systems with Applications , vol.37, No.2, pp1065-1070, 2010.
[18] F. Q. Zhao, Y. Hong, D. M. Yu, etc., "A hybrid particle swarm optimisation algorithm and fuzzy logic for process planning and production scheduling integration in holonic manufacturing systems", International Journal of Computer Integrated Manufacturing, vol.23, No.1, pp20-39, 2010.
[19] Y. W. Guo, W. D. Li, A. R. Mileham, etc., "Optimisation of integrated process planning and scheduling using a particle swarm optimisation approach", International Journal of Production Research, vol.47, No.14, pp3775-3796, 2009.
[20] G. H. Zhang, X. Y. Shao, P. G. Li, etc., "An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem", Computers & Industrial Engineering, vol.56, No.4, pp1309-1318, 2009.
[21] A. Manikas, Y.L. Chang, "A scatter search approach to sequence-dependent setup times job shop scheduling", International Journal of Production Research, vol.47, No.18, pp5217-5236, 2009.
[22] M. Saravanan, A. N. Haq, "A scatter search algorithm for scheduling optimisation of job shop problems", International Journal of Product Development, vol.10, No.1-3, pp259-272, 2010.
[23] C. Y. Peng, L. Gao, X. Y. Shao, etc., "General particle swarm optimization algorithm for job-shop scheduling problem", Jisuanji Jicheng Zhizao Xitong/Computer Integrated Manufacturing Systems, CIMS, vol.12, No.6, pp911-917+923, 2006.
[24] G. G. Yen, B. Ivers, "Job shop scheduling optimization through multiple independent particle swarms", International Journal of Intelligent Computing and Cybernetics, vol.2, pp5-33, 2009.
[25] B. Ivers, G. G. Yen, "Job shop optimization through multiple independent particle swarms", in 2007 IEEE Congress on Evolutionary Computation, September 25, 2007 - September 28, 2007. Singapore: Inst. of Elec. and Elec. Eng. Computer Society, 2008.
[26] A. Gomez, R. Cuevas, D. De La Fuete, etc., "New metaheuristic for the Job Shop Scheduling Problem with PSO", in 2008 International Conference on Artificial Intelligence, ICAI 2008 and 2008 International Conference on Machine Learning; Models, Technologies and Applications, MLMTA 2008, July 14, 2008 - July 17, 2008. Las Vegas, NV, United states: CSREA Press, 2008.
[27] M. R. Chen, X. Li, X. Zhang, etc., "A novel particle swarm optimizer hybridized with extremal optimization", Applied Soft Computing, vol.10, No.2, pp367-373, 2010.
[28] L. dos Santos Coelho, "Gaussian quantum-behaved particle swarm optimization approaches for constrained engineering design problems", Expert Systems with Applications, vol.37, No.2, pp1676-1683, 2010.
[29] K. Mahadevan, P. S. Kannan, "Comprehensive learning particle swarm optimization for reactive power dispatch", Applied Soft Computing, vol.10, No.2, pp641-652, 2010.
[30] R. C. Eberhart, Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization", in Proceedings of the 2000 Congress on Evolutionary Computation CEC 00, July 16, 2000 - July 19, 2000. California, CA, USA: IEEE, 2000.
[31] Y. S. Ong, A. J. Keane, "Meta-Lamarckian learning in memetic algorithms", IEEE Transactions on Evolutionary Computation, vol.8, No.2, pp99-110, 2004.
[32] W. J. Xia, Z. M. Wu, "A hybrid particle swarm optimization approach for the job-shop scheduling problem", The International Journal of Advanced Manufacturing Technology, vol.29, No.3-4, pp360-366, 2006.

***Authors**: prof. dr Fu qing Zhao, School of Computer and Communication, Lanzhou University of Technology, Lanzhou, Gansu, China, E-mail: Fzhaoa2000@hotmail.com; dr Jian xin Tang, School of Computer and Communication, Lanzhou University of Technology, Lanzhou, Gansu, China.*