Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Hybrid Particle Swarm Optimization for Stochastic Flow Shop Scheduling With No-wait Constraint

Bo Liu*, **, Ling Wang***, Bin Qian***
Yihui Jin***

** Centre for World Food Studies, Faculty of Economics and Business Administration, VU University Amsterdam, The Netherlands (Tel: +31 (0)20 598 9321; e-mail: liub01@mails.tsinghua.edu.cn)*
*** Center for Chinese Agricultural Policy, Institute of Geographical Sciences and Natural Resource Research, Chinese Academy of Sciences, Beijing 100101, China*
**** Department of Automation, Tsinghua University, Beijing 100084, China*

Abstract: The flow shop scheduling with the no-wait constraint is a typical NP-hard combinatorial optimization problem and represents an important area in production scheduling. In this paper, a class of particle swarm optimization (PSO) approach with simulated annealing (SA) and hypothesis test (HT), namely PSOSAHT is proposed for the stochastic flow shop scheduling with no-wait constraint to minimize the maximum completion time (makespan). The developed algorithm not only applies evolutionary search guided by the mechanism of PSO, but it also applies the local search guided by the jumping mechanism of SA. Thus, both global exploration and local exploitation are balanced. Meanwhile, it applies HT to perform a statistical comparison to avoid some repeated search to some extent. Simulation results and comparisons demonstrate the feasibility, effectiveness and robustness of the proposed hybrid PSO-based algorithm.

1. INTRODUCTION

Flow shop scheduling problem (FSSP) is a class of widely studied scheduling problems with a strong engineering background, which represents nearly a quarter of manufacturing systems, assembly lines, and information service facilities in use nowadays (Pinedo, 2002). In this paper, the no-wait FSSP is considered, which arises from specific requirements of the production process or from the absence of adequate storage capacity between the operations of a job. In the no-wait FSSP, a job must be processed from start to completion, without any interruption either on or between machines. According to the research (Rock, 1984), the no-wait FSSP with more than two machines is NP-hard.

As we know, in many real manufacturing environments, uncertainty is so prevalent that it is more important and practical to study stochastic scheduling problems than deterministic ones. In this paper, the no-wait FSSP with stochastic processing time is considered where the processing time is assumed as a randomly distributed value. In such a case, the expected makespan is often used to evaluate the performance of the solutions (Pinedo, 2002). Currently, mathematical programming (Luh et al., 1999), constructive methods (Kouvelis et al., 2000), and tabu-search (Yang et al., 2004) have been investigated in scheduling problems with stochastic time.

During recent years, particle swarm optimization (Kennedy and Eberhart, 1995) has gained wide research, which is developed based on observations of the social behaviour of animals, such as bird flocking and swarm theory. PSO is initialized with a swarm of random solutions, where each individual is assigned with a random velocity. According to its own and its companions' flying experiences, each individual (particle) flies through hyperspace. Due to the simple concept, easy implementation, and quick convergence, nowadays, PSO has gained many applications in a variety of fields. However, most studies on PSO are for continuous optimization problems, while little research can be found for combinatorial problems, especially for scheduling problems. Obviously, it is a challenge to employ the algorithm in different areas of problems other than those areas that the inventors originally focused on.

To the best of our knowledge, there is no published work to solve stochastic FSSP with no-wait constraint by using PSO. In this paper, a class of PSO approach with simulated annealing (SA) and hypothesis test (HT), namely PSOSAHT is proposed for no-wait FSSP with stochastic processing time to minimize makespan. Firstly, to make PSO suitable for solving flow shop scheduling, an encoding rule based on random key representation, called ranked-order-value (ROV) rule, is presented to convert the continuous position values of a particle to job permutation. Secondly, the evolutionary searching mechanism of PSO characterized by individual improvement plus population cooperation and competition is utilized to perform the exploration effectively. Thirdly, jumping probability of SA is employed to reduce the probability to be trapped in local minima and to perform the exploitation as well. Moreover, HT is incorporated into the approach to reasonably estimate the solution performance and to reliably identify solution quality so that the repeated search can be reduced to some extent. Simulation results and comparisons demonstrate the feasibility, effectiveness and robustness of the proposed hybrid algorithm.

The remaining contents are organized as follows. In Section 2, the stochastic FSSP with no-wait constraint is introduced. In Section 3, the PSO and HT are introduced respectively. In Section 4, the PSOSAHT is proposed after presenting solution representation, PSO-based search combining HT, and SA-based local search combining HT. In Section 5, experimental results and the effects of some parameters on optimization performance are presented and analysed. Finally, in Section 6, we end the paper with some conclusions and future work.

2. STOCHASTIC FSSP WITH NO-WAIT

The stochastic FSSP with no-wait constraint can be described as follows. Each of n jobs is to be sequentially processed on machine 1, …, m. The processing time $p_{i,j}$ of job i on machine j is supposed to be subjected to a uniform distribution $U((1-\eta)P_{i,j},(1+\eta)P_{i,j})$, where $P_{i,j}$ is the expected processing time, e.g. data provided by the standard benchmarks (Carlier, 1978; Reeves, 1995), and η denotes noise magnitude. At any time, each machine can process at most one job and each job can be processed on at most one machine. The sequence in which the jobs are to be processed is the same for each machine. To satisfy the no-wait restrictions, the completion time of a job on given machine must be equal to the starting time of the job on the next machine. The objective is to find a sequence for processing all the jobs on the machines so that a given criterion is optimized. In the literature, the criterion widely used is the minimization of the maximum completion time, i.e. makespan ($C_{\max}$).

Let $\pi=\{j_1,j_2,...,j_n\}$ denote a permutation of all jobs, and $C(j_i,k)$ denote the completion time of job j_i on machine k, then the completion time $C(j_i,k)$ can be calculated as follows:

$$C(j_1,1)=p_{j_1,1} \tag{1}$$

$$C(j_1,k)=C(j_1,k-1)+p_{j_1,k},k=2,...,m \tag{2}$$

$$C(j_i,k)=C(j_{i-1},1)+\sum_{l=1}^{k}p_{j_i,l}+\Delta(j_i),i=2,...,n,k=1,...,m \tag{3}$$

where

$$\Delta(j_i)=\max\left[0,\max_{2\le k\le m}\left(C(j_{i-1},k)-\left(C(j_{i-1},1)+\sum_{l=1}^{k-1}p_{j_i,l}\right)\right)\right]$$
$$,i=2,...,n \tag{4}$$

Thus, the makespan can be defined as follows:

$$C_{\max}(\pi)=C(j_n,m) \tag{5}$$

The stochastic FSSP with no-wait constraint is then to find a permutation π^* in the set of all permutation Π such that,

$$\pi^*=\arg\{C_{\max}(\pi)=C(j_n,m)\}\to\min\quad\forall\pi\in\Pi \tag{6}$$

3. INTRODUCTION TO PSO AND HT

3.1 PSO

Particle swarm optimization (PSO) is an evolutionary computation technique through individual improvement plus population cooperation and competition, which is based on the simulation of simplified social models, such as bird flocking and the swarm theory. In a PSO system, multiple candidate solutions coexist and collaborate simultaneously. Each solution, called a particle, flies in the searching space to search for the optimal position. A particle, as time passes through its quest, adjusts its position according to its own "experience," as well as the "experience" of its neighbours. Tracking and memorizing the best position encountered, it builds particles' experience. Thus, PSO possesses memory (i.e. every particle remembers the best position that encountered during the past).

The status of each particle in the search space is characterized by two factors: its position and its velocity. The position and the velocity of the *i*th particle in the *d*-dimensional search space are represented as $X_i=[x_{i,1},x_{i,2},...,x_{i,d}]$ and $V_i=[v_{i,1},v_{i,2},...,v_{i,d}]$ respectively. Each particle has its own best position ($pbest$) $P_i=(p_{i,1},p_{i,2},...,p_{i,d})$ corresponding to the personal best objective value obtained so far at time *t*. The global best particle ($gbest$) is denoted by P_g, which represents the best particle found so far at time *t* in the entire swarm. The new velocity of each particle is calculated as follows:

$$\begin{aligned}v_{i,j}(t+1)&=wv_{i,j}(t)+c_1r_1(p_{i,j}-x_{i,j}(t))\\&+c_2r_2(p_{g,j}-x_{i,j}(t)),j=1,2,...,d\end{aligned} \tag{7}$$

where c_1 and c_2 are constants called acceleration coefficients, w is called the inertia factor, and r_1 and r_2 are two independent random numbers uniformly distributed in the range [0, 1].

Thus, the position of each particle is updated in each generation according to the following equation:

$$x_{i,j}(t+1)=x_{i,j}(t)+v_{i,j}(t+1),j=1,2,...,d \tag{8}$$

The particle flies towards a new position according to (7) and (8). This process is repeated until a pre-defined stopping criterion is reached. In addition, the value of each component in V_i can be clamped to the range $[-v_{\max},v_{\max}]$ to control the excessive roaming of particles outside the search space.

3.2 HT

Generally, a stochastic optimization problem can be described as follows:

$$\min_X J(X)=E[L(X,\xi)] \tag{9}$$

where X is a feasible solution of the problem in a finite set and J is the expectation of L, the sample performance as a function of X and ξ (noise or uncertain factors).

Hypothesis test (HT) is an important statistical method that is used to make test for predefined hypothesis based on experiment data (Liu, 1998). To perform HT for two different solutions, it needs multiple independent evaluations to provide suitable performance estimation for decision solutions. If n_i independent simulations are carried out for solution X_i, then its unbiased estimated mean value $\bar{J}_i$ and variance s_i^2 can be calculated as follows:

$$\bar{J}_i = \bar{J}(X_i) = \sum_{j=1}^{n_i} L(X_i, \xi) / n_i \qquad (10)$$

$$s_i^2 = \sum_{j=1}^{n_i} [L(X_i, \xi) - \bar{J}_i]^2 / (n_i - 1) \qquad (11)$$

Considering two different solutions X_1 and X_2, whose estimated performances $\hat{J}(X_1)$ and $\hat{J}(X_2)$ are two independent random variables. According to the law of large number and central limit theorem, the estimation $\hat{J}(X_i)$ subjects to $N(\bar{J}_i, s_i^2 / n_i)$ when n_i approaches to ∞. Suppose $\hat{J}(X_1) \sim N(\mu_1, \sigma_1^2)$ and $\hat{J}(X_2) \sim N(\mu_2, \sigma_2^2)$, where the unbiased estimation values of μ_1, μ_2 and s_1^2, s_2^2 are given by (10) and (11), and let the null hypothesis H_0 be "$\mu_1 = \mu_2$" and the alternative hypothesis H_1 be "$\mu_1 \neq \mu_2$".

When $\sigma_1^2 = \sigma_2^2 = \sigma^2$ and σ^2 is unknown, the critical region of H_0 is described as follows:

$$\left|\bar{J}_1 - \bar{J}_2\right| \geq t_{\alpha/2}(n_1 + n_2 - 2) \cdot Y_1 \cdot Y_2 = \tau \qquad (12)$$

where $Y_1 = \sqrt{[(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2] / (n_1 + n_2 - 2)}$, $Y_2 = \sqrt{(n_1 + n_2)/(n_1 n_2)}$.

Thus, if $\left|\bar{J}_1 - \bar{J}_2\right| < \tau$, i.e. the null hypothesis holds, then it can be regarded that the performances of the two solutions have no significant difference in statistical sense; otherwise, they are significantly different. Furthermore, for uncertain minimization problem it is assumed that X_2 is better than X_1 if $\bar{J}_1 - \bar{J}_2 \geq \tau$, while X_1 is better than X_2 if $\bar{J}_1 - \bar{J}_2 \leq -\tau$. In addition, for a specific problem it often supposes that the theoretical performance variances of all solutions are the same (Liu, 1998), so the hypothesis test can be made according to (12). For a multi-modal stochastic optimization problem, the blind searching with a comparison under HT can often be trapped into local optima. So, it motivates us to propose a hybrid approach by incorporating HT into PSO together with simulated annealing strategy.

4. PSOSAHT FOR STOCHASTIC FSSP WITH NO-WAIT

4.1 Solution representation

Due to the continuous character of the position of particles in PSO, a standard encoding scheme of PSO cannot be directly adopted for the no-wait FSSP. So, the most important issue to apply PSO for the FSSP with no-wait constraint is to find a suitable mapping between the job sequence and the position of particles in PSO. In this paper, a ranked-order-value (ROV) rule based on random key representation (Bean, 1994) is presented to convert the continuous position of particles in PSO $X_i = [x_{i,1}, x_{i,2}, ..., x_{i,n}]$ to the permutations of jobs $\pi = \{j_1, j_2, ..., j_n\}$, thus, the performance of a particle can be evaluated. In our ROV rule, the smallest position value of a particle is firstly handled and assigned a smallest rank value 1. Then, the second smallest position value will be assigned rank value 2. In the same way, all the position values will be dealt with so as to convert the position information of a particle to a job permutation.

For example, consider an instance with 6 jobs, and the position information is $X_i = [0.06, 2.99, 1.86, 3.73, 1.88, 0.67]$. Because $x_{i,1} = 0.06$ is the smallest position value of the particle, $x_{i,1}$ is handled first and assigned the rank value of 1, then the second smallest value $x_{i,6} = 0.67$ is assigned the rank value of 2. In the same way, $x_{i,3}$, $x_{i,5}$, $x_{i,2}$, and $x_{i,4}$ are assigned the rank values 3, 4, 5, and 6, respectively. Thus, the job permutation is obtained, i.e, $\pi = [1,5,3,6,4,2]$.

In our PSOSAHT, a SA-based local search is not directly applied to the position information, but to job permutation. So, when a local search is completed, the particle's position information should be adjusted to guarantee that the permutation converted by the ROV rule for the new position is the same as the permutation obtained by the local search. That is to say, applying the local search approach to job permutation, the position information should be adjusted correspondingly. Fortunately, it is very simple because the process based on some local search methods to position information can be the same as the process to permutation. For example, in Fig. 1, if the SWAP operator is used as a local search operator for job permutation, obviously, the swap of job 5 and job 6 corresponds to the swap of the position values 2.99 and 3.73.

Dimension *j*	1	2	3	4	5	6
Position value	0.06	***2.99***	1.86	***3.73***	1.88	0.67
Job permutation	1	**5**	3	**6**	4	2

Fig. 1. Swap-based local search for job permutation and the corresponding adjustment for position information

4.2 Population initialization

In the standard PSO, the initial swarm is often generated randomly. To guarantee an initial population with a certain quality and diversity, a population initialization procedure is proposed based on the NEH heuristic (Nawaz *et al.*, 1983).

We propose the following initialization procedure: the NEH heuristic is applied to generate the first solution, and then the modified NEH heuristic is used to generate $B\%$ solutions of the initial swarm, while the rest of particles are randomly generated in a certain continuous interval. The modified NEH heuristic works as follows: randomly choose two elements of the job permutation generated by NEH and insert the back one before the front one.

Since the NEH heuristic and its modification result in job permutations, they should be converted to the position values of certain initial particles to perform the PSO-based search. The conversion is performed as follows:

$$x_{NEH,j} = x_{\min,j} + \frac{x_{\max,j} - x_{\min,j}}{n} \cdot \left(s_{NEH,j} - 1 + rand\right), \quad j = 1,2,\ldots,n \tag{13}$$

where $x_{NEH,j}$ is the position value in the *j*th dimension of the particle, $s_{NEH,j}$ is the job index in the *j*th dimension of the permutation by NEH or its modification, $x_{\max,j}$ and $x_{\min,j}$ are the upper and lower bounds of the position value, respectively, *rand* demotes a random number uniformly distributed in interval [0, 1], and *n* represents the number of dimensions of a position which is equal to the number of jobs. Fig. 2 provides an example of the above conversion from job permutation to position information. Obviously, such a conversion obeys the ROV rule. That is, the permutation can be obtained with the position by using the ROV rule.

Dimension *j*	**1**	**2**	**3**	**4**	**5**	**6**
Permutation by NEH	2	4	3	6	5	1
Position value	0.97	2.01	1.88	3.63	3.08	0.53

Fig.2. Conversion of NEH solution (job permutation) to particle position

4.3 PSO-based search with HT

In this paper, PSO-based search, i.e. (7) and (8), is applied for exploration. That is, the position values of particles in current swarm are adjusted by using PSO-based searching operator. Note that, PSO-based evolution is performed on continuous space. So, when evaluating the performance of a particle, the position information should be converted to job permutation by using the ROV rule. On the other hand, aimed at the uncertainty in stochastic optimization problems, multiple independent evaluations should be used to provide a reasonable estimation for the solution. Meanwhile, hypothesis test is added into the PSO to identify the quality of different solutions when updating *pbest* and *gbest*.

4.4 SA-based local search with HT

Simulated annealing (Kirkpatrick *et al.*, 1983) is a famous meta-heuristic, which is of the ability to avoid being trapped in local minima by probabilistic jumping. Starting from an initial state, the system is perturbed at random to a new state in the neighborhood of the original one, for which a change of ΔE in the objective function value takes place. For minimization problems, the new state is accepted with probability $\min\{1, \exp(-\Delta E / T)\}$, where T is a control parameter corresponding to the temperature in the analogy.

In this paper, we design a SA-based local search to enrich the local searching behaviour and to avoid premature convergence. Meanwhile, the HT is applied to identify the quality of solution and to reduce repeated search to some extent. The procedure of SA-based Local Search combining HT can be described as follows:

Step 1: Let the permutation of jobs π_1 be the initial solution, and calculate its performance $\bar{J}_1$ and variance estimation s_1^2 through certain times of independent evaluations. And set π_1 be the best initial solution, i.e. $\pi^* = \pi_1$, and denote the performance estimation and variance estimation of π^* as $\bar{J}^* = \bar{J}_1$ and $s^{*2} = s_1^2$ respectively.

Step 2: Generate a neighbor π_2 of π_1, and estimate its performance $\bar{J}_2$ and variance s_2^2 by certain times of independent evaluations.

Step 3: Perform HT for π_1 and π_2. If the null hypothesis holds, then go back to step 2 to generate another neighbor; otherwise, continue the following steps.

Step 4: Replace the current solution π_1 by π_2 with probability $\min\{1, \exp[-(\bar{J}_2 - \bar{J}_1)/t]\}$, and update π^*, $\bar{J}^*$ and s^{*2} if possible.

Step 5: If M_1 neighbors have been sampled at current temperature, then output the best solution π^*; otherwise, go to step 2.

In this paper, the SWAP operator is utilized in SA-based local search as the neighbor generator. And we set an initial temperature by trail and error. Moreover, exponential cooling schedule, $t_k = \lambda t_{k-1}$ (where $0 < \lambda < 1$), is applied, which is often believed to be an excellent cooling recipe. To provide a rather good compromise between solution quality and search efficiency, the step of Metropolis sampling process is set to $M_1 = n \cdot (n-1)$, where n denotes the number of jobs. In addition, SA-based local search is only applied to the best solution found so far, i.e. *gbest*.

4.5 PSOSAHT

Based on the proposed ROV rule, population initialization, PSO-based search with HT, and SA-based local search with HT, the framework of PSOSAHT for the stochastic FSSP with the no-wait constraint is proposed. The procedure of the PSOSAHT is described as follows:

Step 1 (Initialization)

Step 1.1: Generate $p_s \times B\%$ permutations by NEH and convert them to position values of particles X_i, ($i = 1, \ldots, p_s \times B\%$) according to (13);

Step 1.2: Generate the rest $p_s \times (1-B\%)$ particles X_i with random position values, and determine the corresponding job permutations by ROV rule;

Step 1.3: Evaluate initial population and determine $pbest$ and $gbest$, set initial temperature t_0, and let $g=1$.

Step 2: Repeat until the evaluation value of $gbest$ keeps fixed at consecutive L iterations:

Step 2.1: For each particle i, Perform PSO-based search with HT to update the $pbest$ of each particle and $gbest$;

Step 2.2: For $gbest$, perform the SA-based local search with HT and then update the $gbest$ by using HT;

Step 2.3: Update the temperature and set $g = g+1$;

Step 3: Output the best solution related to $gbest$ particles.

It can be seen that the PSOSAHT not only applies the PSO-based search to effectively perform exploration within the entire region, but it also applies problem-dependent local search to perform exploitation in sub-regions. Meanwhile, HT is applied to identify the quality of solution and to help to reduce repeated search.

5. SIMULATION RESULTS AND COMPARISONS

5.1 Experimental setup

To test the performance of the proposed PSOSAHT for the stochastic no-wait FSSP, 10 well-studied problems that contributed to the OR-Library are selected. The first eight problems are called car1, car2 through car8 by (Carlier, 1978). The other 2 problems are called rec01, and rec03 by (Reeves, 1995), who used them to compare some meta-heuristics and found them particularly difficult. Thus far, these problems have been used as benchmarks for study with different methods by many researchers. Here, the processing time is supposed to be subjected to a uniform distribution $U((1-\eta)P_{i,j},(1+\eta)P_{i,j})$, where $P_{i,j}$ is the expected processing time provided by the above benchmarks, and η denotes noise magnitude.

The PSOSAHT was coded in MATLAB 7.0 and the experiments were executed on a Mobile Pentium IV 2.2 GHz processor with 512MB RAM. In our PSOSAHT, we use the following parameters: the swarm size $p_s = 20$, $B = 25$, $w = 1.0$, $c_1 = c_2 = 2.0$, $x_{\min} = 0$, $x_{\max} = 4.0$, $v_{\min} = -4.0$, $v_{\max} = 4.0$, initial temperature $t_0 = 3.0$, annealing rate $\lambda = 0.9$, stopping parameter $L = 5$, and 10-evaluation is used for solution performance estimation.

5.2 Simulation results and comparisons

To show the effectiveness of PSOSAHT, we carry out simulations to compare our PSOSAHT with PSOSA1 (PSOSAHT without HT element and only one-evaluation is used for solution performance estimation) and PSOSA2 (PSOSAHT without HT element and 10-evaluation is used for solution performance estimation). The noise magnitude is $\eta = 5\%$ for all the problems. Each approach is independently run 20 times for every problem, and the statistical results are listed in Table 1. The values of C^* for car1 through car8 are the optimal makespans or the lower bounds, while the values of C^* for rec01 and rec03 are provided by the RAJ heuristic (Rajendran, 1994). Besides, BEM and AEM denote the relative percentage error of the best and average expected makespan (calculated with the expected processing time for those solutions obtained by the algorithm with estimated performances) with respect to the value C^*, respectively.

From Table 1, it is shown that PSOSAHT is very effective for solving stochastic flow shop scheduling problems. For the 10 benchmark problems, the solutions resulted by PSOSAHT are always the best among the three methods, while PSOSA1 performs the worst. The effectiveness of PSOSAHT is due to the statistical comparison based on hypothesis test to reduce repeated search, reliable performance estimation based on multiple evaluations, and the reasonable combination of global search and local search.

Table 1. Comparisons between PSOSAHT, PSOSA1, and PSOSA2

Problem	*n, m*	PSOSAHT		PSOSA1		PSOSA2	
		BEM	AEM	BEM	AEM	BEM	AEM
car1	11, 5	0	0.11	0	0.51	0	0.18
car2	13, 4	0	0.56	0	0.71	0	0.59
car3	12, 5	0.27	0.66	0.27	1.38	0.27	0.66
car4	14, 4	0.79	5.11	1.02	4.11	1.86	3.52
car5	10, 6	0	1.62	0	1.86	0	1.82
car6	8, 9	0	0	0	0	0	0
car7	7, 7	0	0.17	0	0.28	0	0.10
car8	8, 8	0	0.02	0	0.54	0	0.27
rec01	20, 5	-3.58	- 1.82	-3.21	-1.70	-3.71	- 1.88
rec03	20, 5	-5.28	- 3.55	-4.19	-2.07	-4.19	- 1.84

In addition, the estimation accuracy of objective value is relative to two aspects, i.e., evaluations times for performance estimation and uncertainty magnitude. So, we carry out some simulations on different noise magnitude and different evaluation times. The statistical results are listed in Table 2.

It can be concluded from Table 2 that the more evaluation times are used for performance estimation, the better results can be obtained. Secondly, under the same evaluation times used, PSOSAHT can find better solutions as the uncertainty magnitude decreases. This phenomenon is due to the fact that hypothesis test can work more effectively if estimation is more accurate and performance estimation can be more accurate if uncertainty is smaller. However, more evaluation times may increase the computational effort of the algorithm. Thus, a trade off should be made between the computational effort and the solution quality.

6. CONCLUSIONS

To the best of our knowledge, this is the first report on the application of particle swarm optimization (PSO) approach to the stochastic flow shop scheduling problem (FSSP) with the no-wait constrain with the objective to minimize the maximum completion time. In this paper, a class of PSO approach combining simulated annealing and hypothesis test was proposed for the stochastic flow shop scheduling with no-wait constraint. The developed algorithm not only applied evolutionary search guided by the mechanism of PSO, but it also applied the local search guided by the jumping mechanism of SA. Thus, both global exploration and local exploitation were balanced. Meanwhile, it also utilized HT to perform a statistical comparison to reasonably estimate solution performance and to reliably identify solution quality. By using statistical comparisons guided by HT, some repeated search could be reduced. Simulation results and comparisons demonstrated the effectiveness of the proposed hybrid approach. The future work is to investigate the applications of PSO for other kinds of stochastic scheduling problems as well as multi-objective scheduling problems.

Table 2. Effect of estimation accuracy on PSOSAHT

Probl em	C^*	$\eta = 0.05$			$\eta = 0.10$		
		10-eval.	20-eval.	30-eval.	10-eval.	20-eval.	30-eval.
car1	8142	0.11	0.13	0	0.12	0.12	0.12
car2	8242	0.56	1.07	0.46	1.43	1.36	0.67
car3	8866	0.66	0.49	0.48	1.50	1.46	1.49
car4	9195	5.11	4.76	4.36	7.02	5.18	5.41
car5	9159	1.62	1.47	1.50	3.42	1.98	2.04
car6	9690	0	0	0	0.77	0	0.08
car7	7705	0.17	0.24	0.08	0.49	0.20	0.17
car8	9372	0.02	0.38	0	0.58	0.84	0.48
rec01	1590	-1.82	-2.08	-2.64	0.27	-0.13	-1.07
rec03	1457	-3.55	-3.19	-3.32	-1.32	-0.22	-2.24

ACKNOWLEDGEMENTS

This paper is based on my PhD thesis in Tsinghua University "Study on Optimization and Scheduling Theory and Algorithms Based on PSO for Complex Systems". I am full gratitude of my supervisors, Professor Jin YH and Wang L. I also want to express my sincere gratitude to Professor Keyzer M.A (SOW-VU) and Professor Huang JK (CCAP-CAS) for their support. This research is partially supported by National Natural Science Foundation of China (60774082).

REFERENCES

Bean, J.C. (1994). Genetic algorithms and random keys for sequencing and optimization. *ORSA Journal on Computing*, **6**, 154-160.

Carlier, J. (1978). Ordonnancements a Contraintes Disjonctives. *R.A.I.R.O. Recherche operationelle / Operations Research*, **12**, 333-351.

Kennedy, J. and R.C. Eberhart (1995). Particle swarm optimization. In: *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942-1948.

Kirkpatrick, S., C.D. Gelat, Jr. and M.P. Vecchi (1983). Optimization by simulated annealing. *Science*, **220**, 671-680.

Kouvelis, P., R.L. Daniels and G. Vairaktarakis (2000). Robust scheduling of a two-machine flow shop with uncertain processing times. *IIE Transactions*, **32**, 421-432.

Liu, X. (1998). *Introduction to Statistics*, Tsinghua university Press, Beijing.

Luh, P.B., C. Dong and L.S. Thaku (1999). An effective approach for job-shop scheduling with uncertain processing requirements. *IEEE Transactions on Robotics and Automation*, **15**, 328-339.

Nawaz, M., E. Enscore and I. Ham (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*, **11**, 91-95.

Pinedo, M. (2002). *Scheduling: Theory, Algorithms and Systems,* Prentice-Hall, New Jersey.

Rajendran, C. (1994). A no-wait flowshop scheduling heuristic to minimize makespan. *Journal of The Operational Research Society*, **45**, 472-478.

Reeves, C.R. (1995). A genetic algorithm for flowshop sequencing. *Computers and Operations Research,* **22**, 5-13.

Rock, H. (1984). The three-machine no-wait flowshop problem is NP-complete. *Journal of The Association for Computing Machinery*, **31**, 336-345.

Yang, T., Y. Kuo and I. Chang (2004). Tabu-search simulation optimization approach for flow-shop scheduling with multiple processors -- a case study. *International Journal of Production Research*, **42**, 4015-4030.