# A Hybrid Particle Swarm Optimization Algorithm HPTS for the Flow-Shop Scheduling Problem

Zhang, Xue-Feng
Department of Informatics, Ph.D. Student

Tong, Bin
Graduate School of Systems Life Sciences, Ph.D. Student

Koshimura, Miyuki
Department of Informatics

Fujita, Hiroshi
Department of Informatics

他

https://doi.org/10.15017/18985

出版情報：九州大学大学院システム情報科学紀要. 15 (2), pp.65-69, 2010-09-24. 九州大学大学院システム情報科学研究院
バージョン：
権利関係：

九州大学大学院
システム情報科学紀要
第15巻　第2号　平成22年9月

Research Reports on Information Science and
Electrical Engineering of Kyushu University
Vol.15, No.2, September 2010

# A Hybrid Particle Swarm Optimization Algorithm HPTS for the Flow-Shop Scheduling Problem

Xue-Feng ZHANG*, Bin TONG**, Miyuki KOSHIMURA***, Hiroshi FUJITA*** and Ryuzo HASEGAWA***

(Received July 23, 2010)

**Abstract:** This paper proposes a hybrid particle swarm optimization algorithm for solving Flow-Shop Scheduling Problems (FSSP) to minimize the maximum makespan. A new hybrid heuristic, based on Particle Swarm Optimization (PSO), Tabu Search (TS) and Simulated Annealing (SA), is presented. PSO combines local search (by self-experience) with global search (by neighboring experience), achieving a high search efficiency. TS uses a memory function to avoid being trapped at a local minimum, and has emerged as an effective algorithmic approach for the FSSP. This method can also be referred to as calculation of the horizontal direction. SA employs certain probability to avoid becoming trapped in a local optimum and the search process can be controlled by the cooling schedule (also known as calculation of vertical direction). By reasonably combining these three different search algorithms, we develop a robust, fast and simply implemented hybrid optimization algorithm HPTS (Hybrid of Particle swarm optimization, Tabu search and Simulated annealing). This hybrid algorithm is applied to the standard benchmark sets and compared with other approaches. The experimental results show that the proposed algorithm could obtain the high-quality solutions within relatively short computation time. For 7 of 30 instances, new upper bounds among the unsolved problems are found in a short time in HPTS.

**Keywords:** Flow-shop scheduling problem, Particle swarm optimization, Tabu search, Simulated annealing

## 1. Introduction

The scheduling problem is a simplified model of actual assembly line job shop scheduling problem. Moreover, the research has a very high theory value and a practical application value. Flow-Shop Scheduling Problem (FSSP) is a combinatorial optimization problem and NP-complete[1]. The main task of FSSP is to find a permutation schedule which minimizes the maximum completion time of a sequence of $N = \{J_1, \cdots J_n\}$ jobs in an $M$-machine flow-shop. Every job has $M$ operations, and every machine has $N$ jobs. Every job executes its operations on every machine in the order of $M = \{M_1, \cdots M_m\}$. Every operation of a job cannot be preempted. Every machine can execute only a single operation of a job at a time[2]. This problem can be given as follows:

$$C(1,1) = T(1,1) \tag{1}$$

$$C(1,i) = C(1,i-1) + T(1,i) \tag{2}$$

$$C(r,1) = C(r-1,1) + T(r,1) \tag{3}$$

$$C(r,i) = \max(C(r,i-1), C(r-1,i)) + T(r,i) \tag{4}$$

where $T(r,i)$ denotes the execution time of the $r$th operation of the $i$th job on machine $M_r$, and $C(r,i)$ represents the maximum completion time of the $i$th job on machine $M_r$, $1 \leq r \leq m$, and $1 \leq i \leq n$.

In the past few decades, this problem has attracted many researchers. Many heuristic algorithms have been proposed, such as Taillard's tabu search method[3], Ogbu and Smith's simulated annealing algorithm[4], Reeves's genetic algorithm[5] and Lian et al.'s particle swarm optimization algorithm[6]. Particle swarm optimization (PSO) is an evolutionary computation technique developed by Dr. Eberhart and Dr. Kennedy in 1995, which is inspired by social behavior of bird flocking. It is endowed with properties of easy implementation and fast convergence. In recent years, there have been a lot of reported works focused on the modification of PSO to solve continuous optimization problems[7]. So far, only a few papers have been delivered to solve the FSSP by PSO algorithm. Their experimental results show that they are more effective than the algorithms based on Genetic Algorithms (GA) and constructive heuristics. However, these algorithms are still suffering from the problem of premature convergence and being easily trapped into local optimum. This is mainly caused by the decreasing degree of swarm diversity that leads to a total implosion and ultimately fitness stagnation of the swarm[8]. In this paper, we focus on exploiting

*Department of Informatics, Ph.D. Student
**Graduate School of Systems Life Sciences, Ph.D. Student
***Department of Informatics

particle swarm optimization algorithm to get the solutions for FSSP. We propose a hybrid particle swarm optimization algorithm for solving FSSP to minimize the maximum makespan. A new hybrid heuristic, based on Particle Swarm Optimization (PSO), Tabu Search (TS) and Simulated Annealing (SA), is presented. By reasonably combining these three different search algorithms, we develop a robust, fast and simply implemented hybrid optimization algorithm HPTS.

The following paper is organized as follows: PSO is described in Section 2. Section 3 discusses and analyzes the structure and description of the HPTS algorithm. Section 4 gives experimental results of the HPTS algorithm and other competitive approaches. Finally, the main conclusions are drawn.

## 2. Particle swarm optimization

The particle swarm concept was based on the premise of social behavior. The original intent was to graphically simulate the graceful but unpredictable choreography of a bird flock. A PSO algorithm mimics the behavior of flying birds and their means of information exchange to solve optimization problems. PSO has been introduced as an optimization technique in real-number spaces. But many optimization problems are set in a space featuring discrete components. Typical examples include problems that require ordering and route planning, such as in scheduling and routing problems[9]. They are described as follows[10]:

$$V_{id} = \omega \times V_{id} + C_1 \times \mathrm{Rand}() \times \left(\mathrm{P}_{\mathrm{id}}^{\mathrm{best}} - \mathrm{P}_{\mathrm{id}}\right) + C_2 \times \mathrm{Rand}() \times \left(\mathrm{P}_{\mathrm{gid}}^{\mathrm{best}} - \mathrm{P}_{\mathrm{id}}\right) \quad (5)$$

$$P_{id} = P_{id} + V_{id} \quad (6)$$

where $V_{id}$ representes the velocity of particle $i$. It also can be regarded as the distance to be traveled by particle $i$ from its current position. $P_{id}$ represents the particle position, $P_{id}^{best}$ called "pbest", the local best solution, represents particle $i$'s best previous position, and $P_{gid}^{best}$ called "gbest", the global best solution, represents the best position among all particles in the swarm. $\omega$ is an inertia weight. It regulates the trade-off between the global exploration and local exploitation abilities of the swarm. The acceleration constants $C_1$ and $C_2$ represent the weights of the stochastic acceleration terms that pull each particle toward "pbest" and "gbest" positions. Rand() is a random function with range $[0, 1]$.

For Eq.5, the first part represents the inertia of previous velocity; the second part is the "cognition" part, which represents individuals thinking independently; and the third part is the "social" part, which represents cooperation among the particles[11].

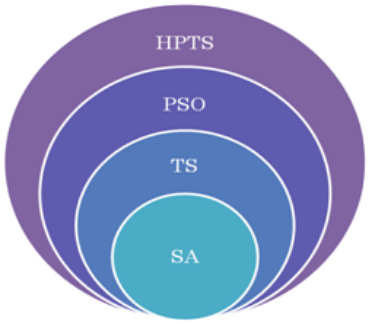


**Fig. 1** The structure of HPTS.

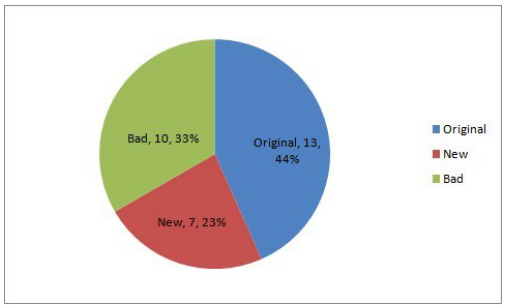


**Fig. 2** The detailed comparison of experimental results

## 3. HPTS

In the HPTS algorithm, the groups use a random initialization, which generates initial particle's initial position and velocity in the search space. Each particle's best position is set to the current location, and the current position of the corresponding fitness value is calculated for each particle, according to the evolutionary structure. Figure 1 shows the structure of HPTS.

Our algorithm consists of three steps. The first step is initialization, which is presented as follows.

### 3.1 PSO

(1-1) Initialize swarm, including swarm size, each particle's position and velocity;
(1-2) Evaluate each particle's fitness;
(1-3) Initialize *gbest* position with the particle having the lowest fitness in the swarm;
(1-4) Initialize *pbeset* position with a copy of particle itself;
(1-5) Give an initial value.

### 3.2 TS[*1]

(2-1) Generate an initial solution $s$ and calculate its makespan $f$(s), set the current solution $s$=$s*$, the best solution $pbest = s$, the makespan $f$(pbest) = $f$(s), $iter = 0$ and push the $s*$ onto the elite solution

[*1] TS includes SA

stack $L$ (LIFO list);

(2-2) Determine initial temperature $T_0$, and the termination $T_{end}$.

The second step is a computation which consists of two sub-steps, PSO and TS, respectively. PSO is shown in algorithm 1.

**Algorithm 1** PSO

1: **while** the maximum of generation is not met **do**
2: Generation++;
3: Generate next swarm by Eq. 5 and 6;
4: Find new *gbest* and *pbest*;
5: Update *gbest* of the swarm and *pbest* of each particle;
6: **end while**

TS and SA are shown in algorithm 2.

**Algorithm 2** TS (SA)

1: Set $iter = iter + 1$, generate neighbors of the current solution $s*$ by a neighborhood structure. If the $s*$ is optimal, then stop;
2: Select the best neighbor which is not tabu or satisfies the aspiration criterion, and store it as the new current solution $s*$, update the tabu list;
3: **for** *gbest* particle $s$ of swarm **do**
4: $T_k = T_0$;
5: **while** $T_k \geq T_{end}$ **do**
6: Generate a neiborhood solution $s*$ from $s$ by pair-exchange method;
7: Compute fitness of $s*$;
8: Evaluate s*, $\Delta = f(s^*)\text{-}f(s)$;
9: **if** min[1,exp(-$\Delta/T_k$)] < random[0,1] **then**
10: Accept $s*$;
11: Update the best solution found so far if possible;
12: **end if**
13: **end while**
14: $T_k$= B * $T_{k-1}$;
15: **end for**
16: If $iter \leq improveiter$ then go to loop;
17: If a termination criterion is satisfied then stop. otherwise 'pop' a solution on top of the solution stack $L$, shift the solution to active schedule and install the active solution as the current solution s*, set $iter = 0$, and empty tabu list. Go to step 2;

Step3. outputs the optimization results.

## 4. Experimental results

In this section, three steps including PSO, TS and SA are considered. In addition, a repulsive process, which is able to make the particles fly toward some promising areas to search some un-reached regions, is well incorporated into HPTS. The effect of swarm activity strategy on the performance of HPTS and the new multi-layer hybrid particle swarm optimization model are firstly discussed. Note that the comprehensive experimental evaluations and comparisons of the proposed algorithms with NPSO and GA are given in 8).

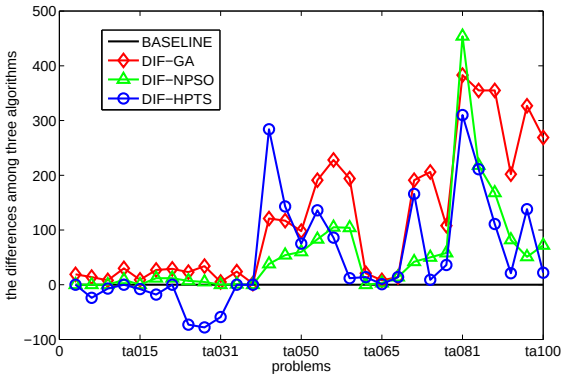


**Fig. 3** The comparison of three algorithms.

To illustrate the effectiveness of the HPTS algorithm for FSSP to minimize the makespan, 30 instances of 10 different sizes taken from Taillard's benchmark (Taillard, 1990) have been selected for simulation experiments. This benchmark contains some instances that have been proven to be very difficult to solve in the sense that the best solutions found so far are through the use of a very lengthy Tabu-search heuristic.

In HPTS, the population size is 100, which means we use only 100 particles to search solutions for every problem where the number of iterations is 400. In order to compare the convergence rate with GA and NPSO, we run the HPTS algorithm 10 times for every problem. The proposed two algorithms were implemented in MATLAB and simulated in a platform with 2.27 GHz Intel(R) Core(TM)2 Duo CPU, RAM 4GB.

In **Table 1**, $PS$ denotes the problem's size with the number of jobs ($Js$) times the number of machines ($Ms$), $FSSP$ represents the Flow-shop Scheduling Problem, $WK$ represents the well-known optimal solutions in the benchmark, $Min$, $Max$ and $Avg$ represent the best solution, the worst solution, and the average solution over 10 times, respectively. *Original* represents the some results equal to $WK$. *New* indicates the best solutions better than $WK$. and *Bad* means the bad results worse than $WK$. The detailed comparison of experimental results are shown in **Fig. 2**.

In **Fig. 3**, these parameters satisfy the following conditions: $DIF-HPTS = HPTS$ - $WK$; $DIF-NPSO = NPSO$ - $WK$; $DIF-GA = GA$ - $WK$; $BASELINE = WK$ - $WK$; From **Fig. 4**, we can observe the gantt charts of each experimental results in HPTS, which include ta001 – ta091 of 10 different scale experimental

**Table 1** The experimental results.

| PS(Js*Ms) | FSSP | WK | NPSO | GA | HPTS |
|---|---|---|---|---|---|
| | | | Min Avg Max | Min Avg Max | Min Avg Max |
| 20*5 | ta001 | 1278 | 1278/1295.1/1297 | 1297/1297.0/1297 | 1278/1278.0/1278 |
| 20*5 | ta005 | 1236 | 1236/1247.9/1250 | 1250/1251.2/1258 | **1212**/1229.7/1247 |
| 20*5 | ta010 | 1108 | 1108/1115.4/1127 | 1116/1135.6/1161 | **1101**/1107.5/1114 |
| 20*10 | ta011 | 1582 | 1591/1604.7/1627 | 1612/1623.7/1645 | 1582/1585.7/1589 |
| 20*10 | ta015 | 1419 | 1419/1428.1/1444 | 1428/1457.4/1478 | **1411**/1415.0/1419 |
| 20*10 | ta020 | 1591 | 1603/1621.0/1633 | 1618/1632.8/1654 | **1573**/1582.3/1591 |
| 20*20 | ta021 | 2297 | 2309/2324.8/2339 | 2326/2344.8/2375 | 2297/2297.0/2297 |
| 20*20 | ta025 | 2291 | 2298/2312.2/2334 | 2314/2330.6/2351 | **2218**/2255.0/2291 |
| 20*20 | ta030 | 2178 | 2183/2215.1/2244 | 2212/2237.5/2282 | **2100**/2139.0/2178 |
| 50*5 | ta031 | 2724 | 2724/2729.5/2742 | 2729/2739.6/2752 | **2665**/2694.5/2724 |
| 50*5 | ta035 | 2863 | 2864/2864.0/2864 | 2887/2916.3/2947 | 2863/2863.0/2863 |
| 50*5 | ta040 | 2782 | 2782/2783.2/2786 | 2784/2815.7/2832 | 2782/2782.5/2783 |
| 50*10 | ta041 | 3025 | 3063/3098.6/3138 | 3146/3186.7/3227 | 3309/2724.0/3366 |
| 50*10 | ta045 | 2986 | 3040/3078.7/3129 | 3103/3168.4/3194 | 3129/3221.0/3311 |
| 50*10 | ta050 | 3091 | 3151/3171.7/3204 | 3189/3225.5/3280 | 3166/3170.0/3174 |
| 50*20 | ta051 | 3875 | 3958/3991.5/4011 | 4066/4105.3/4141 | 4011/4143.5/4276 |
| 50*20 | ta055 | 3635 | 3740/3773.7/3812 | 3863/3915.9/3975 | 3721/3803.5/3886 |
| 50*20 | ta060 | 3777 | 3881/3959.8/4034 | 3971/4008.6/4093 | 3789/3895.0/4001 |
| 100*5 | ta061 | 5493 | 5493/5494.0/5495 | 5514/5524.5/5541 | 5507/5509.8/5512 |
| 100*5 | ta065 | 5250 | 5253/5256.8/5267 | 5258/5302.5/5336 | 5251/5252.0/5253 |
| 100*5 | ta070 | 5328 | 5342/5345.8/5368 | 5342/5372.3/5403 | 5342/5342.0/5342 |
| 100*10 | ta071 | 5770 | 5812/5842.8/5869 | 5961/6027.2/6095 | 5936/6080.5/6225 |
| 100*10 | ta075 | 5468 | 5518/5578.4/5636 | 5674/5769.2/5850 | 5477/5497.5/5518 |
| 100*10 | ta080 | 5845 | 5903/5914.5/5962 | 5953/6056.3/6106 | 5881/5892.0/5903 |
| 100*20 | ta081 | 6286 | 6470/6544.4/6615 | 6669/6763.9/6843 | 6596/6834.0/7072 |
| 100*20 | ta085 | 6377 | 6595/6653.6/6723 | 6732/6816.3/6898 | 6588/6731.0/6874 |
| 100*20 | ta090 | 6465 | 6633/6723.3/6821 | 6820/6910.3/6979 | 6576/6824.0/7072 |
| 200*10 | ta091 | 10868 | 10950/10978.9/11005 | 11070/11112.1/11168 | 10889/11017.0/11145 |
| 200*10 | ta095 | 10524 | 10575/10664.0/10764 | 10851/10917.7/11029 | 10662/10721.5/10781 |
| 200*10 | ta100 | 10676 | 10748/10796.9/10850 | 10945/11025.6/11101 | 10698/10735.5/10773 |

data.

**Table 1** shows the experimental results of different benchmark instances. The comparison of the three different algorithms is shown in **Fig. 3**. According to the values of $Min$, $Max$ and $Avg$, and the curve about GA, NPSO, and HPTS, we have an observation that HPTS outperforms the NPSO and GA algorithms in the total solution quality thoroughly. Specific experimental results including ta001 – ta091 of 10 different scale experimental data are described in **Fig. 4**, which illustrate respectively the effectiveness and performance of HPTS for FSSP to minimize makespan proposed from the gantt charts of each experimental results.

What is more important is shown in **Fig. 2**. HPTS is capable of achieving competitive results compared to $WK$ in 20 cases out of 30 instances. That is 67% of experimental results are acceptable. It is worthy of noting that, in the 67% portion, the results for 7 instances, i.e. ta005, ta010, ta015, ta020, ta025, ta030 and ta031, are much better than those of $WK$. It demonstrates the HPTS algorithm has a powerful exploring ability.

Figure 3 shows the comparison of HPTS with GA and NPSO. It can be observed obviously that our HPTS algorithm outperforms the NPSO and GA algorithms in some solution quality thoroughly (In particular, ta005, ta010, ta015, ta020, ta025, ta030 and ta031). HPTS can achieve the optimal value in the search space quickly with smaller population size, making use of its better local searching ability to get the final optimal solution at the end of evolution.

Therefore, the computational results show that the proposed algorithm could obtain the high-quality solutions within relatively short computation time. So, it is a robust, fast and simply hybrid optimization algorithm.

## 5. Conclusions

Although there are many literatures on classical PSO, the PSO for FSSP does not have rich literatures. In this paper, a new hybrid heuristic HPTS, based on Particle Swarm Optimization (PSO), Tabu Search (TS) and Simulated Annealing (SA), is presented. We have applied the HPTS algorithm to FSSP in minimizing

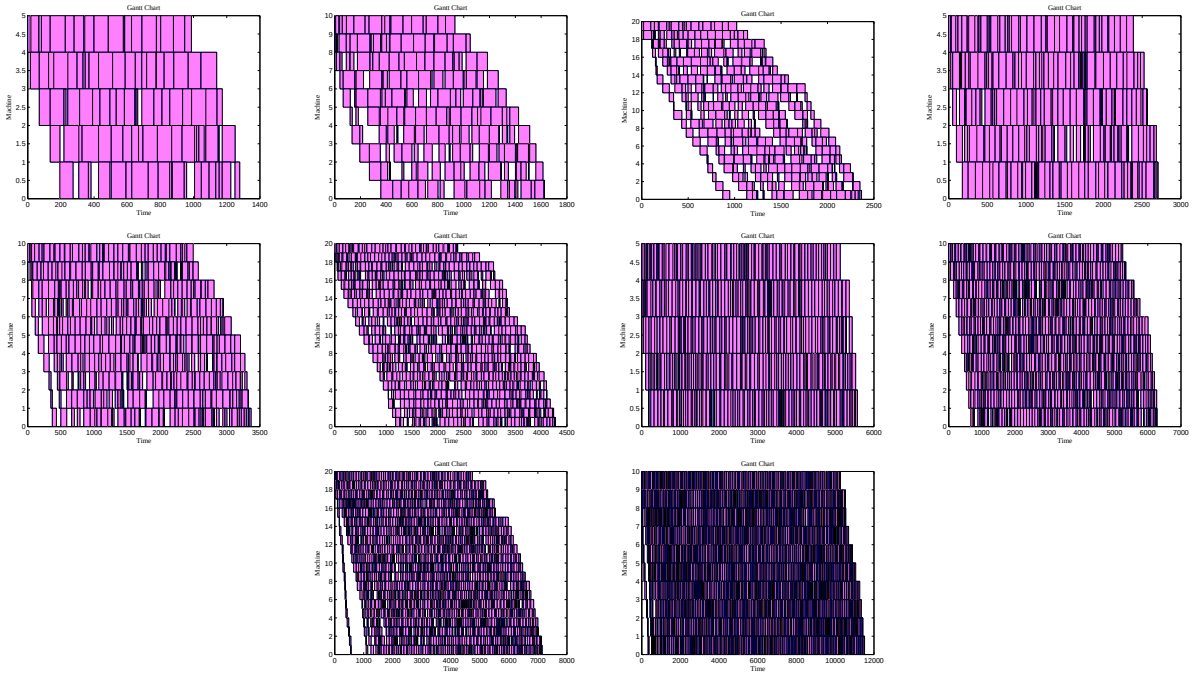


**Fig. 4** The gantt charts of each experimental results in HPTS.

makespan. The experimental results show the effectiveness of the proposed approach. Moreover, our HPTS algorithm outperforms the NPSO and GA algorithms in some solution quality thoroughly. For 7 of 30 instances, new upper bounds among the unsolved problems are found in a short time in HPTS. It can achieve the optimal value in the search space quickly with smaller population size, making use of its better local searching ability to get the final optimal solution at the end of evolution.

In additon, we have proposed a new algorithm structure: three-tier structure of hybrid algorithm, which brings the new research direction and inspiration for the hybrid algorithm. To improve the algorithm performance, we plan to apply parallel and distributed computing to the HPTS algorithm. Applying HPTS to other NP problems would also be promising. We will also give the mathematical demonstration of HPTS.

## Acknowledgments

This paper is supported by the research fund KAKENHI (20240003, 21300054), Xue-Feng Zhang and Bin Tong are sponsored by the China Scholarship Council (CSC).

## References

1) Garey,M.,Johnson,D.Sethi,R. The complexity of flow shop and job shop scheduling. Mathematics of Operations Research, 1, 117-129, 1976.
2) I-Hong Kuo, Shi-Jinn Horng. An efficient flow-shop scheduling algorithm based on a hybrid particle swarm optimization model. Expert Systems with Applications 36, 7027-7032, 2009.
3) Taillard,E. Some efficient heuristic methods for the flow shop sequencing problem. Euopean Journal of Operational Research, 47(1), 65-74, 1990.
4) Ogbu,F.A., Smith, D. K. The application of the simulated annealing algorithm to the solution of the n/m/Cmax flow shop problem. Computers and Operations Research, 17, 243-253, 1990.
5) Kennedy, J., Eberhart, R. Particle swarm optimization. In Proceedings of IEEE international conference on neural network , 1942-1948, 1995.
6) Lian, Z., Gu, X., Jiao, B. A novel particle swarm optimization algorithm for permutation flow-shop scheduling to minimize makespan. Chaos, Solitons and Fractals, 35, 851-861, 2008.
7) He, S., Wu, Q. H., Wen, J. Y., Saunders, J. R., Paton, R.C. A particle swarm optimizer with passive congregation. BioSystems, 78, 135-147, 2004.
8) Changsheng Zhang, Jiaxu Ning , Dan tong Ou yang. A hybrid alternate two phases particle swarm optimization algorithm for flow shop scheduling problem. Computers & Industrial Engineering, 58, 1-11, 2010.
9) D. Y. Sha, Hsing-Hung Lin. A particle swarm optimization for multi-objective flowshop scheduling. ORIGINAL ARTICLE Int J Adv Manuf Technol, 45, 749-758, 2009.
10) Shi Y, Eberhart R. Empirical study of particle swarm optimization. In: Proceedings of Congress on Evolutionary Computation, 1945-1950, 1999.
11) Kennedy J. The particle swarm: social adaptation of knowledge. In: Proceedings of IEEE International Conference on Evolutionary Computation, 303-308, 1997.