# Objective Flow-Shop Scheduling Using PSO Algorithm

**Dhahir A. Abdullah**
collage of science\computer department

## Abstract

Swarm intelligence is the study of collective behavior in decentralized and self-organized systems. Particle swarm optimization algorithm (PSOA) models the exploration of a problem space by a population of agents or particles. In this paper, PSOA is used to reduce the makespan and idle time of jop-shop scheduling problem. The proposed algorithm update the speed ($v_i^k$ ) and position ($X_i^k$ ) depend on local (Pbest ) and global (Gbest ) values, in order to find best solutions. The critical path is found by drawing Gantt chart.

**Keywords:** makespan, PSO-practice swarm algorithm, job scheduling problem.

## المستخلص

يعتبر ذكاء السرب دراسة للسلوك الجماعي في الأنظمة غير المركزية وذاتية التنظيم .خوارزمية مفاضلة السرب الجزئية تستخدم فضاء المشكلة من خلال جيل الجزئيات. في هذا البحث تم استخدام خوارزمية مفاضلة السرب الجزيئية لتقليل مديات وفترات التوقف لمشكلة جدولة اعمال المتجر. الخوارزمية المقترحة تقوم بتحديث السرعة والموقع بالاعتماد على القيم المحلية الخارجية من اجل الحصول على افضل النتائج ولقد تم ايجاد المسار الحرج من خلال رسم جنات.

## Introduction

Particle swarm optimization (PSO) is an evolutionary technique for unconstrained continuous optimization problems proposed by Kennedy et al. [1] The PSO concept is based on observations of the social behavior of animals such as birds in flocks, fish in schools, and swarm theory. To minimize the Objective of maximum completion time(i.e.,….the makes pan), Liu et al. [2] invented an effective PSO-based mimetic algorithm for the permutation flow shop scheduling problem. Rajendran et al. [3] developed a PSO algorithm for solving the permutation flow shop scheduling problem.

Most studies of flow shop scheduling have focused on a single objective that could be optimized independently. However, empirical scheduling decisions might not only involve the consideration of more than one objective, but also require minimizing the conflict between two or more objectives. In addition, finding the exact solution to scheduling problems is computationally expensive. Solving a scheduling problem with multiple objectives is even more complicated than solving a single-objective problem.

Rajendran et al. [3] approached the problem of scheduling in permutation flow shop using two ant colony optimization (ACO) approaches, first to minimize the makespan, and then to minimize the sum of the total flow time. Yagmahan [4] was the first to apply ACO meta-heuristics to flow shop scheduling with the multiple objectives of makespan, total flow time, and total machine idle time. The new algorithm is based on the principle of particle swarm optimization (PSO). PSO as an evolutionary algorithm, it combines coarse global search capability (by neighboring experience) and local search ability. Eren et al. [5] tackled a multi-criteria two-machine flow shop scheduling problem with minimization of the weighted sum of total completion time, total tardiness, and makespan.

***The aim of this research*** is to explore the development of PSO for elaborate multi-objective flow-shop scheduling problems. The original PSO was used to solve continuous optimization problems. Due to the discrete solution spaces of scheduling optimization problems, we modified the particle position representation, particle movement, and particle velocity in this research.

The remainder of this research contains a formulation of the flow-shop scheduling problem with two objectives, describes the algorithm of the proposed PSO approach, the random number generation to find p best, G best, velocity, and position and find the sequence of Job and draw the grant and find the total completion time (critical bath time to achieve all Jobs).

## The Fitness and Critical Path

There are three ways of swarm , the ***first*** , that compute the fitness of each particle and find the optimal of all that means Pbest , and the max neighbor is Gbest , and update the velocity and the position of particle to find the solution , the **second** find the fitness and the optimal fitness and the update the velocity by using generate random number . The **third** one, or the new method is find the fitness with probability of some factor[6] .

The critical path is the longest path. There may be several critical paths in one disjunctive graph, and the length of each critical path exactly equals the makespan. Any operation on the critical path is called a critical operation. A critical operation cannot be delayed without increasing the makespan of the schedule. It is possible to decompose the critical path into a number of blocks. A block is a maximal sequence of adjacent critical operations processed on the same machine. The first and the last operations of this block are called block head and block rear, respectively. Other operations are called internal operations [7].

## The Proposed Algorithm

The proposed algorithm is described in algorithm(1)

Algorithm (1) the proposed algorithm

| |
|---|
| Step 1: generate the processing time by $t_{(i,j)} = -\frac{1}{\lambda} \log(1-r)$ , ƛ ..... average processing time, r...... random number. |
| Step 2: find the machine sequence there are $(n!)^m$ state solution. |
| Step 3: generate random number to find the initial permutation, position ($X_i^k$), velocity ($V_i^k$) and find P $best^k$ of G $best^k$ . |

Step(4) can be described by following:

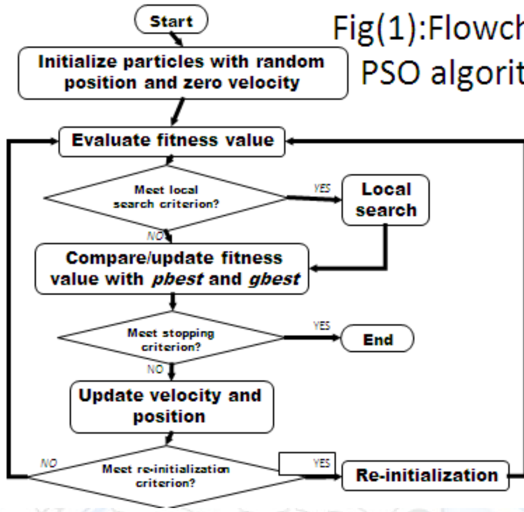
update the velocity and position as follows.

1. Find $P_{best}$ is the smallest value of fitness "variance" if f( $x_{i(t)}$) < $P_{best}$ then P best e=F($X_{i(t)}$) and X $P_{best\ i}$ = Xi(t) and $P_{best}$ is the current best fitness achieved by with particle and X $P_{best\ i}$
2. If F($X_{i(t)}$)< $L_{besti}$ the $L_{best\ i}$ : F($X_{i(t)}$) where $L_{best\ i}$ fitness over the topological neighbors "the smallest value".
3. Update the velocity $V_i(t)$ and the position (Xi(t) )of each particle.

$$V_i(t)= V_i(t-1)+ r_1 (X\ P\ best_i - X\ i(t) )+ r_2 (X\ L\ best_i - X\ i(t) ) \qquad (1)$$

$$\text{And} \quad X_i(t)= X_i(t)+ V_i(t) \qquad (2)$$

Where, r1, r2 random number




Fig(1):Flowchart of PSO algorithm

The problem of scheduling in flow shops has been the subject of much investigation. The primary elements of flow shop scheduling include a set of *m* machines and a collection of *n* jobs to be scheduled on the set of machines. Each job follows the same process of machines and passes through each machine only once. Each job can be processed on one and only one machine at a time, whereas each machine can process only one job at a time. The processing time of each job on each machine is fixed and known in advance. We formulate the multi-objective flow shop scheduling problem using the following notation:

- A set of n jobs are scheduled on a set of m machines.
- Each of consists of a set of operations which their machine orders are pre-specified.
- The required machine and the fixed processing time characterize each operation.
- $t(_{i,j})$ is the processing time for job $i$ on machine $j$ ($i$=1,2,…n) and ($j$=1,2,…m), and

- Start time of each operation
- End time of each operation.
- Solution space=$(n!)^m$

The objectives(*fitness*) considered in this research can be calculated as follows:

Makespan( max*fCmCn)=summation of time processing of jobs in critical path*

*Machine idle time* (machine don't work).

The problem n*m problem size 3 jobs *4 machines (as shown in table (1)):

1. Generate the processing time by using exponential distribution at

Lim =3 (the mean average time processing)

$$t(i,j) = -\frac{1}{\lambda}\log(1-r) \qquad (3)$$

r………..random number (0-1).

Table: (1) nxm problem size: 3 jobs x 4 machines.

| Job | Machine Sequence | Processing Time/Day |
|---|---|---|
| 1 | M1 M2 M4 M3 | 3 3 5 2 |
| 2 | M4 M1 M2 M3 | 4 1 2 3 |
| 3 | M2 M1 M3 M4 | 3 2 6 3 |

PSO for job scheduling problem:

- Particle Representation
- Random key.
- Initially the value in each position is randomly generated.
- Subsequent values are defined via the position update equation defined previously.

Particle Representation

- Random key.

- initially the value in each position is randomly generated.
- Subsequent values are defined previously.

2. Generate the random number for each machine as follows:

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.13 | 0.21 | 0.23 | 0.45 | 0.29 | 0.32 | 0.09 | 0.46 | 0.36 | 0.39 | 0.25 | 0.18 |

3. Arrange the random number and sequence in order.

| 7 | 1 | 12 | 2 | 3 | 11 | 5 | 6 | 9 | 10 | 4 | 8 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.09 | 0.13 | 0.18 | 0.21 | 0.23 | 0.25 | 0.29 | 0.32 | 0.36 | 0.39 | 0.45 | 0.46 |

4. Put the number of job for each order sequence.

| 7 | 1 | 12 | 2 | 3 | 11 | 5 | 6 | 9 | 10 | 4 | 8 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 |

5. Rearrange for the first sequential we get the Pbest.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 3 | 2 | 2 | 1 | 3 | 3 | 3 | 2 | 1 |

**P best= [ 1 1 2 3 2 2 1 3 3 3 2 1 ]**

6. Repeat step 2-5 to estimate the Gbest :

(a). Generate the random number for each machine:

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.13 | 0.25 | 0.41 | 0.51 | 0.61 | 0.44 | 0.32 | 0.54 | 0.76 | 0.62 | 0.91 | 0.11 |

(b). Arrange the random number as follows:

| 12 | 1 | 11 | 2 | 7 | 3 | 6 | 4 | 8 | 5 | 10 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.11 | 0.13 | 0.19 | 0.25 | 0.32 | 0.41 | 0.44 | 0.51 | 0.54 | 0.61 | 0.62 | 0.76 |

(c). Put the number of job for each order sequence as flows:

| 12 | 1 | 11 | 2 | 7 | 3 | 6 | 4 | 8 | 5 | 10 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 |

(d). Rearrange for the first sequential we get the Gbest.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 2 | 2 | 3 | 2 | 2 | 3 | 3 | 3 | 1 | 1 |

**G best= [1 1 2 2 3 2 2 3 3 3 1 1]**

7. To generate the velocity $V_i$ randomly we used the following form:

If R<0.33 then v=-1

If R>0.33-0.66 then v=0

If R>0.66 then v=1

We get the result from run the program:

R1=0.09, R2=0.11, R3=0.71, R4=0.55, R5=0.75, R6=0.21

R7=0.01, R8=0.42, R9=0.77, R10=0.04, R11=0.76, R12=0.74

$V_i^k$= [ -1 -1 1 0 1 -1 -1 0 1 -1 1 1]

8. To estimate the $X_i$ we used the formula as in[5].

$X^K$= [ 1.4 1.61 3.21 4.05 0.15 1.71 2.51 3.92 0.14 1.54 3.26 4.24].

$Pbest^K$ = [1 1 2 3 2 2 1 3 3 3 2 1]

9. Update the position ($x_i$)

and velocity($v_i$) as follows:

T=1 iteration 1

J1 v1≠0 x1=x1+v1 then x1=0.4, v1=-1

J2 v2≠0 x2=x2+v2 then x2=0.61, v2=-1

J3 v3≠0 x3=x3+v3 then x3=4.21, v3=1

T=2 iteration 2

J1 v4=0 since rand1=0.6, rand2=0.3, rand1>c1+c2 , where c1=0.7, c2=0.8 random number

X4=pbest4+rand2-0.5 x4=2.8, v4=-1

J2 v5≠0 x5=x5+v5 x5=1.15, v5=1

J3 v6≠0 x6=x6+v6 x6=0.71, v6=-1

T=3 iteration 3

J1 v7≠0 x7=x7+v7 x7=1.51, v7=-1

J2 v8=0 rand1=0.85, rand2=0.7, rand1>c1+c2

x8=gbest+rand2-0.5 x8=3.2 v8=-1

J3 v9≠0 x9=x9+v9 x9=1.14, v9=1

T=4 iteration 4

J1 v10≠0 x10=x10+v10 x10=0.54, v10=-1

J2 v11=0 rand1=0.85, rand2=0.7, rand1>c1+c2

X11, v11 do not change x11=3.26, v11=1

J3 v12=0 rand1=0.95, rand2=0.7, rand1>c1+c2

X12, v12 do not change x12=4.24, v12=1

We get the new position and velocity

X=[ 0.4 0.61 4.21 2.8 1.15 0.71 1.51 3.2 1.14 0.54 3.26 4.24 ]

V=[ -1 -1 1 -1 1 -1 -1 0 1 -1 1 1]

10. Sequence of jobs using Small Positive Value (spv).

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.4 | 0.61 | 4.21 | 2.8 | 1.15 | 0.71 | 1.51 | 3.2 | 1.14 | 0.54 | 3.26 | 4.24 |

| 1 | 10 | 2 | 6 | 9 | 5 | 7 | 11 | 4 | 8 | 3 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.4 | 0.54 | 0.61 | 0.71 | 1.14 | 1.51 | 1.51 | 2.8 | 3.2 | 3.26 | 4.21 | 4.24 |

| 1 | 10 | 2 | 6 | 9 | 5 | 7 | 4 | 8 | 11 | 3 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 3 | 3 | 3 | 3 |

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 3 | 2 | 2 | 1 | 2 | 3 | 3 | 1 | 2 | 3 |

**G best=[ 1 1 3 2 2 1 2 3 3 1 2 3 ]**

11. Draw the Gantt chart of the Gbest as shown in Fig(2):

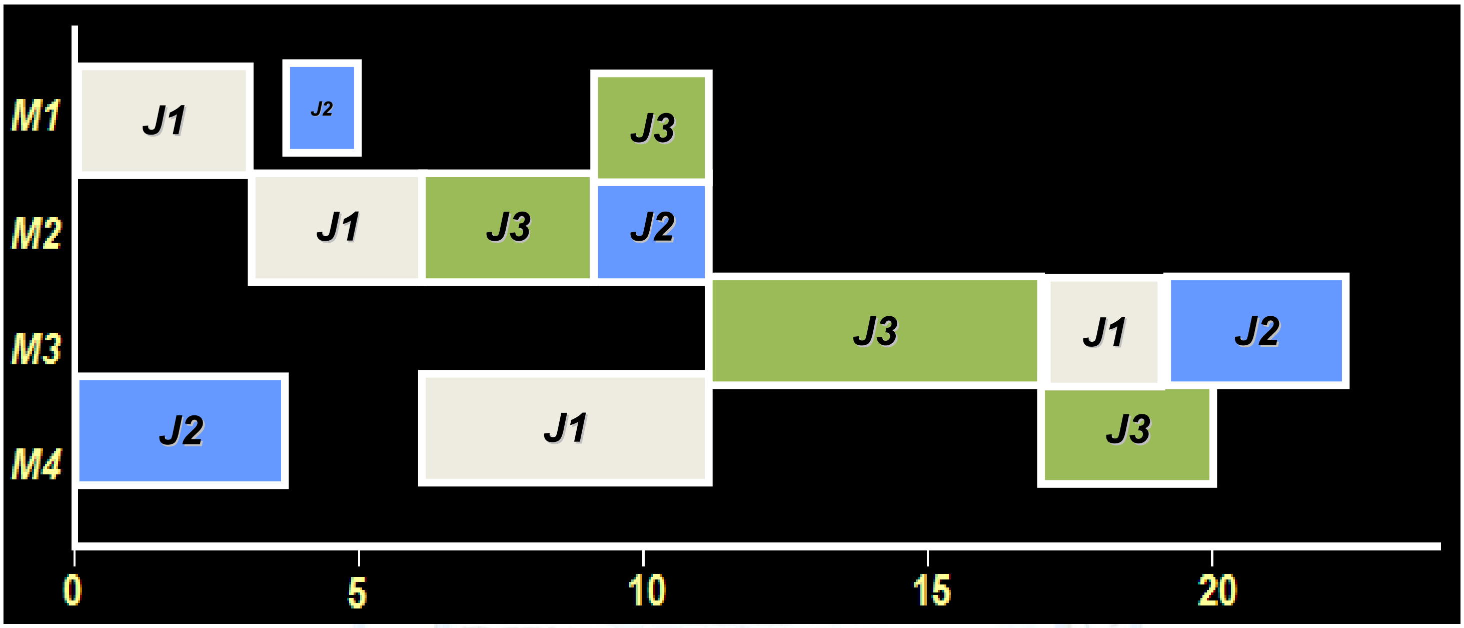
**Idle time= 32**

**Maskespan= 22 day**

**Fitness= 0.6 (maskespan) + 0.4 (Idle time) =0.6*22+0.4*32=26**

**Critical path of jobs J1 J1 J3 J2 J3 J1 J2 = 22**

Fig(2): Gbest chart.




G best position

Fig(2): Gbest chart.

G best position

Permeation:

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

**G best:**

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **1** | **1** | **3** | **3** | **2** | **1** | **2** | **3** | **2** | **1** | **2** | **3** |

**<u>Jop sequence</u>**

| **Group of jobs** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|
| Job 1 : | $M_{1,1}$ [0, 3 ] | $M_{2,2}$ [3, 6 ] | $M_{4,6}$ [6, 9 ] | $M_{3,10}$ [17, 19 ] |

| | | | | |
|---|---|---|---|---|
| Job 2 : | $M_{4,5}$ [0, 4 ] | $M_{1,7}$ [4, 5 ] | $M_{2,9}$ [9, 11 ] | $M_{3,11}$ [19, 22 ] |
| Job 3 : | $M_{2,3}$ [6, 9] | $M_{1,4}$ [9, 11 ] | $M_{3,8}$ [11, 17 ] | $M_{4,12}$ [17, 20 ] |

Since: M $_{4,12}$ [17, 20] number of machine is 4, permutation of the job is 12.

The star time of job is 17 and the end time of job is 20.

**Machine sequence**

| | | | | |
|---|---|---|---|---|
| M1 | Job $_{1,1}$ [3] | Job $_{2,2}$ [5] | Job $_{3,2}$ [11] | **end time [11]** |
| M2 | Job $_{1,2}$ [6] | Job $_{3,1}$ [9] | Job $_{2,3}$ [11] | **end time [11]** |
| M3 | Job $_{3,3}$ [17] | Job $_{1,4}$ [19] | Job $_{2,4}$ [22] | **end time [29]** |
| M4 | Job $_{2,1}$ [4] | Job $_{1,3}$ [9] | Job $_{3,4}$ [20] | **end time [20]** |

Since : Job 3,4 [20] number of Job is 3, number of group of Job is **4**, and the end time of Job is 20.

## Results

To illustrate the effectiveness and performance of algorithm proposed (the fitness) in this research, three representative instances (*represented by problem nXm*) based on practical data have been selected to compute. Three problem instances (problem 3X4, problem 8X8 and problem 10 X 10) , the mathlap program was used to solve the problems, the complete time is 22, 75, 73 respectively .

Also we can exchange the position of J3 on machine M2 with neigh bow Job1 in sequence or Job2, then the position of Job3 on M1, M3 and M4 change that minimize the make span the critical path and idle time and give good fitness.

G best$^2$, when change J3 with J1, the best position is:

1 **3** **1** 3 2 1 2 3 2 1 3 2

G best[3], when change J3 with J2 on M2 the best position is:

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | **2** | **3** | 3 | 1 | 2 | 3 | 2 | 1 | 2 | 3 |

Then repeat the Gant diagram and find the fitness for G best[2]

M1 end time **( 7 )**

M2 end time **( 8 )**

M3 end time **( 18 )**

M4 end time **( 17 )**

**Fitness = 0.6*18+0.4*18 = 18**

**Critical path = 18**

The calculations are showing in the following steps:

1. calculate the variance as follows:

$$Fi = \sum_{i=1}^{12} fi \quad (4)$$

$$\overline{F} = mean = \frac{F}{n} \quad (5)$$

$$\text{Variance} = \frac{1}{12} \sum_{i=1}^{12} (fi - F)^2 \quad (6)$$

2. $X_i$ [1,2,3,4,5,6,7,8,9,10,11,12]

   $V_i$ [22,18, 20, 21, 24, 23,16,17,15, 24, 25, 26]

3. P best $_i$ = 0.07 the smallest value.

   X best =3 the position of Pbest.

4. Also after update x(t) use ( xi(t)= mod 12) this mean the new position from the previous.
5. Repeat the all 4 step for the new position.

| 1 | 2 | 3 | 4 |
|---|---|---|---|
| **22** | **18** | 20 | 21 |
| 24 | 23 | 16 | 17 |
| 15 | 24 | 25 | 26 |

Table 2: the time of each job.

Mean= $\bar{F} = \frac{22+18+20+21+24+23+16+17+15+24+25+26}{12} = 20.9$

From equation (6) calculate the variance as following:

$F_{(X1)} = 0.1008$ $F_{(X4)} = 0.8$

$F_{(X2)} = 0.241$ $F_{(X5)} = 0.36$

$F_{(X3)} = 0.076$ $F_{(X6)} = 2.0$

## Conclusion

Many flow shop scheduling problem studies have been conducted in the past. However, the objective these studies was minimization of the maximum completion time (i.e., the makespan), to improve efficiency and reduce production costs. However, there has been limited study of PSO to address the multiple objectives. Therefore PSO method for solving a flow shop scheduling problem with multiple objectives is presented, including minimization of makespan, mean flow time, and machine idle time(fitness).

PSO algorithm is less time to find the variance. The results of our performance measurement also revealed that the proposed PSO algorithm out performed, the heuristics in minimizing the makespan, mean flow time, and total machine idle time. Comparing with the previous result such as in [3,5,6], we find the smallest variance.

## Reference


1. Kennedy J, Eberhart R (1995), "Particle swarm optimization", Proceedings of the IEEE International Conference on Neural Networks PP.1942–1948.
2. Liu B, Wang L, Jin YH (2007), "An effective PSO-based memetic algorithm for flow shop scheduling", IEEE Vol37:PP.18–27.
3. Rajendran C, Ziegler H (2004), "Ant-colony algorithms for permutation flow shop scheduling to minimize makespan/total flow time of jobs", Europe Journal Operational Research Vol.155: PP.426–438.
4. Yagmahan B, Yenisey MM (2008), "Ant colony optimization for multi-objective flow shop scheduling problem", Computer Engineering Vol.54:PP.411–420.
5. Eren T, Güner E (2007), "The tricriteria flow shop scheduling problem", International Journal Advance Manufacture Technology Vol.36:pp.1210–1220.
6. Pasupathy T, Rajendran C, and Suresh RK (2006), "A multi-objective genetic algorithm for scheduling in flow shops to minimize the makespan and total flow time of jobs". International Journal Advance Manufacture Technology Vol.27:PP.804–815.
7. Lian Z, Gu X, Jiao B (2008), "A novel particle swarm optimization algorithm for permutation flow-shop scheduling to minimize makespan", Chaos, Solutions and Fractals Vol. PP. 35:851–861.