A novel discrete PSO algorithm for solving job shop scheduling problem to minimize makespan

IOP Conf. Series: Materials Science and Engineering **310** (2018) 012143 doi:10.1088/1757-899X/310/1/012143

A novel discrete PSO algorithm for solving job shop scheduling problem to minimize makespan

K Rameshkumar[1, 3], **C Rajendran**[2]

[1]Department of Mechanical Engineering, Amrita School of Engineering ,Bangalore, Amrita Vishwa Vidyapeetham, Amrita University, India -560035
[2]Department of Management studies, Indian Institute of Technology Madras, Chennai, India -600036

E-Mail : ramesh_amrita@yahoo.com

Abstract. In this work, a discrete version of PSO algorithm is proposed to minimize the makespan of a job-shop. A novel schedule builder has been utilized to generate active schedules. The discrete PSO is tested using well known benchmark problems available in the literature. The solution produced by the proposed algorithms is compared with best known solution published in the literature and also compared with hybrid particle swarm algorithm and variable neighborhood search PSO algorithm. The solution construction methodology adopted in this study is found to be effective in producing good quality solutions for the various benchmark job-shop scheduling problems.

1. Introduction

Job-shop scheduling is considered as the hardest combinatorial optimization problem [1]. Scheduling involves allocation of jobs in machines available in the shop-floor. A typical schedule provides information relevant to the timing of the jobs to be loaded on the machines. When the number of jobs and machines increases, the complexity of solving the problem also increases. Scheduling of the jobs over the machines are carried out with respect to a particular objective considering separately or combined together. Based on some of the important objective such as minimization of make-span, minimization of flow-time, and minimizing the tardiness, schedules were generated by the researchers and practicing engineers.

In this study, schedules are generated based on minimizing the make-span criterion. The make-span is defined as total completion time of all the jobs in all the machines in the shop-floor. The reduction of make-span results in improving the throughput. The make-span (C^*_{max}) is computed using the expression given in the equation 1 for an 'n' job 'm' machine problem [2]. Each job 'j_i' in the shop-floor has to undergo 'O' operations utilizing 'm' machines. Jobs are to be scheduled as per the process plans following the precedence and resource constraints. The processing time 'τ_{ik}' of job 'j' in the machine 'm' is provided as an input the algorithm. The starting time 't_{ik}' of the job 'j' is computed

[3] To whom any correspondence should be addressed.

with an objective of minimizing the make-span criterion by following the priority and resource capacity constraints. It is also assumed that revisiting of the jobs are not allowed and a job will be processed only once in a machine.

$$C^{*}_{max} = \min\left(C_{max}\right) = \min_{feasibleschedules}\left(\max\left(t_{ik}+\tau_{ik}\right): \forall\ J_i \in J, M_k \in M\right) \quad (1)$$

Various optimization techniques have been used over the years for solving shop scheduling problem. These techniques includes exact methods, branch and bound algorithms, heuristics based on dispatching rules, Tabu search, Simulated annealing algorithm, Genetic algorithm, Local search techniques, Particle swarm algorithms, Differential evolution algorithm and hybrid techniques. A detailed survey on job-shop scheduling has been extensively carried out by the researches. Mckay et al. [3], Holthaus and Rajendran [4], Yamada et al. [5], Błażewicz et al. [6,7], Jain and Meeran [2], Jones et al. [8], and Wang and Zou [9] have done a detailed survey on job-shops focusing on optimization procedures, mathematical formulations, and future directions. Simulation based approaches for minimizing makespan in a job-shop has been studied by Thenarasu et. al [10].
Particle Swarm Optimization proposed by Kennedy and Eberhart [11] is inspired form the intelligent behaviour of swarm and has been implemented to variety of problems in science and engineering involving continuous and discrete variables. Clerc [12] implemented discrete version of PSO for traveling salesman problems. A continuous version of PSO for permutation flow shop scheduling problems by Tasgetiren et al. [13]. Attempts have been made by Tasgetiren et al. [14] and Xia and Wu [15] to solve job shop scheduling problems using PSO algorithm with the objective of minimizing the makespan. Tasgetiren et al. [13,14] proposed a smallest position value rule (SPV) to enable the continuous particle-swarm optimization algorithm to solve the permutation flowshop scheduling problems and job shop scheduling problems by converting position values to its permutation of operations. Xia and Wu [15] proposed a new hybrid particle swarm optimization algorithm combines a continuous particle swarm algorithm with simulated annealing algorithm. Rameshkumar et al. [16, 17, 18] proposed a PSO algorithms for solving flow-shop scheduling problem. Karthi et al. [19] implemented discrete and continuous version PSO algorithms for data clustering problems. Kadadevaramath et al.[20] proposed an intelligent PSO model for solving supply chain network optimization problem. Sun and Xiong [21] implemented PSO for job shop scheduling. A hybrid Tabu – PSO algorithm was proposed by Gao et al. [22]. Meng et al. [23] fused GA and SA in PSO for improving its performance.

In this work, job-shop scheduling problems are solved using the proposed novel discrete version of PSO algorithm. Optimal / near optimal schedules were generated by considering minimizing the makespan criterion. Solution quality of the proposed algorithm is evaluated by solving bench-mark problem and comparing with best-known solution available in the literature.

2. Particle representation and proposed schedule builder

Operation based representation proposed by Gen et al. [24] is considered in this paper to implement the proposed PSO algorithm for the job shop scheduling problem. All operations for a particular job is represented with discrete values and then inferred it as per the order of incidence. For a '$n \times m$' job-shop problem a particle in PSO is represented by 'nm' integers. For example, for a three job (n), three machine problem (m), the representation would be as shown in figure 1, where 1 stands for job1, 2 stands for job2 and three stands for job3. There will be nine integers to represent a particle in PSO algorithm.

Particle representation	3	2	2	1	1	2	3	1	3
	First Operation for job 3	First Operation for job 2	Second operation for job 2	First Operation for job 1	Second Operation for job 1	Third Operation for job 2	Second Operation for job 3	Third Operation for job 1	Third Operation for job 3

Figure 1. Particle representation

2.1 Schedule Builder

The schedule builder is a module of the evaluation procedure and should be chosen with respect to the performance measure of optimization. Most of the important performance measures of the job shop scheduling problems are regular measures, which mean that optimal solutions are always semi-active [25]. Semi-active scheduling methodology is used to generate optimal solution from the particle representation scheme for the performance criterion such as minimizing the makespan. Computational experiments showed, that minimum makespan job shop scheduling problem improves by the use of a more powerful schedule builder, in particular an active scheduler. An active schedule builder performs a kind of local-search to improve the solution quality. This strategy, first used by Nakano and Yamada [26], is called forcing. Left-shifting strategy is used in the schedule builder to generate the schedule. In this study, we have considered a semi active schedule builder and as well as a novel active schedule builder to construct the solution for the given sequence of operations.

2.2 Proposed active schedule builder

An active scheduler performs left shifts and does the forcing operation. The procedure used to construct the active schedules is based on a scheduling generation scheme that does time incrementing. The algorithmic description of the schedule generation scheme used to generate the active schedules is shown in Figure 2.

Set $t = 0$, $RT_m = 0$ (Release_time of machine m)
Phase I:
For every machine m: do
If $t = (RT)_m$; Release the job on that machine m;
Then
Activate the next operation of that job;
Put it in the corresponding machine queue;
Phase II:
For every machine m: do
If $t \geq (RT)_m$ and machine m is free and queue exists;
Then
Choose one from queue as per order;
Load that operation on machine m;
Update $(RT)_m = t +$ processing time
Go to Phase I until all jobs are scheduled.

Figure 2. Schedule generation scheme

3. Benchmark problems

Bench-mark problems are used test the performance of proposed algorithms. These bench-mark problems were proposed by many researchers. These bench-mark problems were solved by many researchers by various approaches and techniques and reported their results in the literature. The bench-mark problems proposed by researchers are of wide range of sizes and also their difficulty level of solving also varies. There are many problems in the bench-mark where the optimal solutions are not found due to its combinatorial nature. These benchmark problems are formulated by various authors.

To illustrate the effectiveness and performance of the proposed algorithm, we have considered 40 problem instances (LA01 ~ LA40) of eight different sizes due to Lawrence [27], 5 problems (ABZ5 ~ ABZ9) of 2 different sizes due to Adams et al. [28], three problem instances called FT06, FT10 and FT20 due to Fisher and Thompson [29] and 10 problems (ORB1 ~ ORB10) used by Applegate and Cook [30]. We have solved all the 58-benchmark problems using the proposed PSO algorithm and the results were compared with the PSO algorithms proposed by Tasgetiren [13,14] and Xia and Wu [15].

4. Proposed discrete version PSO for job-shop scheduling

4.1 Structure of PSO algorithm for Job- shop scheduling problem

The main objective is to determine the sequence of operations performed on the jobs in the corresponding machines that provides the minimum makespan for the job–shop problem. Make-span is the completion time of all the operation of all the jobs in all the machines. A sequence consisting of 'nm' integers. No. of jobs is represented as 'n' and no. of machines as 'm'. Each integer in the sequence represents the sequence of operations performed for the jobs in the machines. In PSO, a particle means a sequence.

Solution construction in PSO is taking place by considering current position of the particle $\{C_{s,n}\}$, best position reached by the particle in a particular point of time during the evolution $\{B_{s,n}\}$ and overall best position among all the particles $[G_n]$. After construction of new sequence, $\{C_{s,n}\}$, $\{B_{s,n}\}$ and $\{G_n\}$ are updated by calculating the makespan of the sequence. The sequence is represented by the letter's' and 'n' is the number of dimensions. The proposed version of PSO is not similar to the generic PSO used for solving continuous function optimization problems. The structure of the proposed discrete PSO is given as follows:

Step1: Generate the sequences randomly.
No. of sequences (Swarm Size) Say 'S'. i.e., $\left(C_{s,n}, n = 1, 2, ..., N\right)$ for $s = 1, 2, \ldots, S$.

Step2: Generate the feasible sequence proposed by Gen et al. [24].

Step3: For all $\{C_{s,n}\}$, find the make-span of the sequence (objective function value) $\left(\text{i.e.,} f\left(\left\{C_{s,n}\right\}\right)\right)$. [Active schedule builder and Semi active schedule builder are used for construction the schedule]

Step4: Initialize $\{B_{s,n}\}$ and $\{G_n\}$

Step5: While (No. of iterations / No. of solutions generated is not reached)
do for each particle
{
build a new sequence (particle);
}

Step6: Update $\{C_{s,n}\}$, $\{B_{s,n}\}$ and $\{G_n\}$

Step 6 : Report the best sequence / solution found.

4.2 Building a new sequence

A new sequence of the particle '*s*' is built using the current position of the particle is $\{C_{s,n}\}$, best position reached by the particle say at an iteration 't' is $\{B_{s,n}\}$ and over-all best position among all the particles is $\{G_n\}$. $\{B_{s,n}\}$ and $\{G_n\}$ are identified based on the make-span value arrived by evaluating the sequences (particles).

A random number in the range U[0,1] is generated and compared with the set weights w_c, w_b and w_g. The set weights w_c, w_b and w_g corresponds to the sequences (particle) $\{C_{s,n}\}$, $\{B_{s,n}\}$ and $\{G_n\}$ for building the new sequence. The weights are generated in such a way that $w_c+w_b+w_g=1$.

IOP Conf. Series: Materials Science and Engineering **310** (2018) 012143 doi:10.1088/1757-899X/310/1/012143

The building of a new sequence $\left\{C_{s,n}^{new}\right\}$ is initialized form the null sequence (at the start there are no jobs in the sequence). At the start, The new sequence $\left\{C_{s,n}^{new}\right\}$ is set to 'ϕ,' a null sequence, is as follows.

For s = 1 (1) *n* :

{

sample *u ;* U [0,1];

if $u \le w_c$

then

assign the 1st job from $\{C_{s,n}\}$, which is not scheduled yet in $(C_{s,n}^{new})$, and add it to $\left\{C_{s,n}^{new}\right\}$.

else

if $\left(u \le w_b + w_g\right)$

then

assign the 1st job from $\{B_{s,n}\}$ which is not scheduled yet in $(C_{s,n}^{new})$, and add it to $(C_{s,n}^{new})$.

else [if $\left(w_c + w_b\right) < u \le 1$]

assign the 1st job from $\{G_n\}$ which is not scheduled yet in $(C_{s,n}^{new})$, and add it to $(C_{s,n}^{new})$.

}

To demonstrate the sequence building procedure, lt is assumed that that there is a sequence of four operations for a 3- job 3-machine problem (i.e., particle *s*) in the swarm as given below:

$\{C_{s,n}\}$ = {1, 2, 1, 3, 2, 3, 2, 1, 3};

$\{B_{s,n}\}$ = {2, 1, 2, 1, 3, 3, 2, 1, 3}; and

$\{G_n\}$ = {1, 3, 2, 1, 3, 2, 2, 1, 3}.

Let w_c, w_b and w_g are the relative importance, which is similar to the coefficients c_1 and c_2 used in generic PSO equation for calculating velocity as shown in the equation 1. Velocity of the particle is denoted by 'v','t' is the iteration counter, 'c_1' and 'c_2' are the relative significance of the 'social' and 'cognitive' coefficients which is used to find the velocity to update the new position. 'X(t)' is the current position of the particle and 'P_{best}' and 'G_{best}' are particle's best and global best positions. Two uniformly distributed random numbers 'r_1' and 'r_2' are in the range [0,1]. New position is updated after calculating the velocity. New position of the particle is given in the equation 2.

$$v_i(t+1) = v(t^{t-1}) + c_1 r_1 (P_{best}(t-1) - X_i(t)) + c_2 r_2 (G_{best}(t) - X_i(t)) \quad (2)$$

$$X_i(t+1) = X_i(t) + v_i(t+1) \text{, where 'i' = 1,... n} \quad (3)$$

For the demonstration of constructing the new solution, Let us consider $w_c = 0.2$, $w_b = 0.3$ and $w_g = 0.5$ (w_c, w_b and w_g are sampled in the range U[0,1] in a such way that $w_c+w_b+w_g=1$). The building of a new sequence starts with a null set, i.e. $\left\{C_{s,n}^{new}\right\} = \left\{\varphi\right\}$. Corresponding to the sequence of operations, nine random numbers are generated in the range U[0,1]. Let the random numbers are 0.98, 0.45, 0.35, 0.24, 0.64, 0.04, 0.95, 0.17 and 0.10. Corresponding to the random numbers, by following the construction procedure, the new sequence

$\left\{C_{s,n}^{new}\right\}$ arrived as $\{1, 2, 1, 2, 3, 1, 2, 3, 3\}$. In this novel procedure, the velocities used in the conventional PSO are defined with the help of w_c, w_b and w_g. The relative importance for current position is applied through w_c and for Particle's best and Global best is through w_b and w_g respectively.

5. The proposed discrete PSO Algorithm for job-shop scheduling

The notations used in the algorithm are given as follows.

t	itr. number.
t_{max}	termination criteria (Max. number of iterations).
S	size of the swarm.
$\{C_{s,n}\}$	current sequence of a particle.
$f\left(\left\{C_{s,n}\right\}\right)$	value of makespan of sequence.
$\{B_{s,n}\}$	particle's best.
$f(\{B_{s,n}\})$	value of makespan of sequence $\{B_{s,n}\}$.
$\{G_n\}$	global best sequnce
$f(\{G_n\})$	value of makespan of the sequence ξ.
w_c, w_b, w_g	weights assigned to sequence building for $\{C_{s,n}\}$, $\{B_{s,n}\}$ and $\{G_n\}$.
ϕ	null set.
I_s	improved sequence (after implementation of local search).
$f(I_s)$	value of makespan yielded by I_s.
u	a uniform random number generated in the interval U[0,1].

Step1

Set t=1, t_{max} = 50 and S=20. Construct (S) sequences at random. Find the function value (make-span) $f\left(\left\{C_{s,n}\right\}\right)$ of the sequences. Initiate, $w_c = 0.20$, w_b=0.30, and $w_g = 0.50$.

Step2

Apply improvement scheme for every sequence, $\{C_{s,n}\}$. [17]

Indicate the resultant sequence by $\{B_{s,n}\}$.

Let $\{B_{s,n}\}^*$ be the sequence such that $f(\{B_{s,n}\}) = \min\left\{f(\{B_{s,n}\})\right\}$.

Set $\{G_n\}=\{B_{s,n}\}^*$, and $f(\{G_n\}) = f(\{B_{s,n}\}^*)$

For s = 1 to S :

{

Compare$\{C_{s,n}\}$, $\{B_{s,n}\}$ and $\{G_n\}$.

If $\{C_{s,n}\}=\{B_{s,n}\}=\{G_n\}$; construct new sequence randomly and subject it to improvement scheme then generate a new $\{C_{s,n}\}$ randomly, and implement improvement scheme and update $\{B_{s,n}\}$ and $\{G_n\}$ if there is an improvement.

}

Step3

Increment the iteration counter, $t = t+1$.

Step4

Set $\left\{C_{s,n}^{new}\right\}$ = ϕ, a null sequence.

Step5

For all particles, s = 1to S: do the following:

For i= 1 to n do:

{
sample $u \sim U[0,1]$;
if $u \le w_c$
then
assign the 1st job from $\{C_{s,n}\}$, which is not scheduled yet in $(C_{s,n}^{new})$, and add it to $\{C_{s,n}^{new}\}$.
else
if $(u \le w_b + w_g)$
then
assign the 1st job from $\{B_{s,n}\}$ which is not scheduled yet in $(C_{s,n}^{new})$, and add it to $(C_{s,n}^{new})$.
else [if $(w_c + w_b) < u \le 1$]
assign the 1st job from $\{G_n\}$ which is not scheduled yet in $(C_{s,n}^{new})$, and add it to $(C_{s,n}^{new})$.
Set $(C_{s,n}^{new}) = \{C_{s,n}\}$
}

Step6
Do the following:
Step6.1: Apply improvement scheme to $\{C_{s,n}\}$. Say the resultant sequence by I_s.
Let I^* be the sequence where $f(I^*) = \min\{f(I)\}$.
If $f(I^*) < f(G)$
then set $G = I^*$, and $f(G) = f(I^*)$.
Step6.2: For $s = 1$ to S do the following:
{
if $f(\{B_{s,n}\}) > f(I_s)$
then set $\{B_{s,n}\} = I_s$ and $f(\{B_{s,n}\}) = f(I_s)$.
}
Step6.3: For $s = 1$ to S do:
Set $\{C_{s,n}\} = I_s$ and $f(\{C_{s,n}\}) = f(I_s)$.
Step7
for $s = 1$ to S do the following:
{
Compare $\{C_{s,n}\}$, $\{B_{s,n}\}$ and $\{G_n\}$.
If $\{C_{s,n}\} = \{B_{s,n}\} = \{G_n\}$; construct new sequence randomly and subject it to improvement scheme then generate a new $\{C_{s,n}\}$ randomly, and implement improvement scheme and update $\{B_{s,n}\}$ and $\{G_n\}$ if there is an improvement.
}
Step 8
If $t \le t_{max}$ then go to Step3;
else Report solution. The sequence $\{G_n\}$ is the sequence providing minimum make-span and $f(\{G_n\})$ is the make-span of the sequence.

The above algorithm is implemented in visual C++ platform and 58 bench-mark problems were solved and results were compared with the best-known solution available in the literature.

6. Performance analysis of discrete PSO algorithm

The bench-mark problems are solved by the proposed discrete version of the PSO algorithm. Swarm size considered in this study is 20 and max number of iterations allowed to report the results using the proposed algorithm is 50. The make-span reported by the algorithm is indicated in the Table 1. The make-span of the proposed algorithm is compared with best known solution available in the literature and two PSO algorithms proposed by Tasgetiren et al. [13] and Xia and Wu [14]. Tasgetiren et al.[13] hybridized their PSO algorithm with local search with a variable neighbourhood search method. Xia and Wu [14] hybridized their PSO with simulated annealing technique. The relative performance increase (RPI) over the best known solution is also presented in the Table 1-4. The relative percent increase in makespan over the best-known solution is calculated using the equation 3 for the proposed algorithm.

$$\text{Relative percent increase in makespan, } \% = \frac{\text{Makespan}_{\text{D-PSO}} - \text{BKS}}{\text{Makespan}_{\text{D-PSO}}} \times 100 \quad (4)$$

It is observed from the results that for 37 bench-mark problems PSO was able to achieve the best known solution available in the literature. This shows that the proposed discrete version PSO is able to produce good quality solutions for the job-shop scheduling problems. Swarm size considered in this study is 20. Max number of iterations allowed to report the results using the proposed algorithm is 50.

Table 1. Performance of algorithms- Problems provided by Adams et al. [27]

S.No.	Problem	Size (n×m)	BKS	H-PSO	PSO-VNS	D-PSO	$RPI_{D\text{-}PSO}$, %
01	abz5	10 x10	1234	1234	1234	1238	0.32
02	abz6	10 x10	943	943	943	945	0.21
03	abz7	20 x 15	656	666	659	688	4.88
04	abz8	20 x 15	646	681	674	708	9.60
05	abz9	20 x 15	662	694	688	713	7.70

Table 2. Performance of algorithms- Problems provided by Fisher and Thompson [28]

S.No.	Problem	Size (n×m)	BKS	H-PSO	PSO-VNS	D-PSO	$RPI_{D\text{-}PSO}$, %
01	FT06	6 x 6	55	55	55	**55**	0.00
02	FT10	10 x10	930	930	930	938	0.86
03	FT20	20 x 5	1165	1178	1165	**1165**	0.00

Table 3. Performance of algorithms- Problems provided by Applegate and Cook [29]

S.No.	Problem	Size (n×m)	BKS	H-PSO	PSO-VNS	D-PSO	$RPI_{D\text{-}PSO}$, %
01	orb01	10 x10	1059	1059	1059	**1059**	0.00
02	orb02	10 x10	888	889	889	889	0.11
03	orb03	10 x10	1005	1020	1005	1027	2.19
04	orb04	10 x10	1005	1006	1005	1006	0.10
05	orb05	10 x10	887	887	887	**887**	0.00
06	orb06	10 x10	1010	1010	1013	**1010**	0.00
07	orb07	10 x10	397	397	397	**397**	0.00
08	orb08	10 x10	899	899	899	**899**	0.00
09	orb09	10 x10	934	934	934	**934**	0.00
10	orb10	10 x10	944	944	944	**944**	0.00

Table 4. Performance of algorithms- Problems provided by Lawrence [26]

S.No.	Problem	Size (n×m)	BKS	H-PSO	PSO-VNS	D-PSO	RPI_{D-PSO}, %
01	LA01	10 x 5	666	666	666	**666**	0.00
02	LA02	10 x 5	655	655	655	**655**	0.00
03	LA03	10 x 5	597	597	597	**597**	0.00
04	LA04	10 x 5	590	590	590	**590**	0.00
05	LA05	10 x 5	593	593	593	**593**	0.00
06	LA06	15 x 5	926	926	926	**926**	0.00
07	LA07	15 x 5	890	890	890	**890**	0.00
08	LA08	15 x 5	863	863	863	**863**	0.00
09	LA09	15 x 5	951	951	951	**951**	0.00
10	LA10	15 x 5	958	958	958	**958**	0.00
11	LA11	20 x 5	1222	1222	1222	**1222**	0.00
12	LA12	20 x 5	1039	1039	1039	**1039**	0.00
13	LA13	20 x 5	1150	1150	1150	**1150**	0.00
14	LA14	20 x 5	1292	1292	1292	**1292**	0.00
15	LA15	20 x 5	1207	1207	1207	**1207**	0.00
16	LA16	10 x10	945	945	945	**945**	0.00
17	LA17	10 x10	784	784	784	**784**	0.00
18	LA18	10 x10	848	848	848	**848**	0.00
19	LA19	10 x10	842	842	842	**842**	0.00
20	LA20	10 x10	902	907	902	910	0.89
21	LA21	15 x10	1046	1047	1047	1047	0.10
22	LA22	15 x10	927	927	927	**927**	0.00
23	LA23	15 x10	1032	1032	1032	**1032**	0.00
24	LA24	15 x10	935	938	935	939	0.43
25	LA25	15 x10	977	977	977	984	0.72
26	LA26	20 x 10	1218	1218	1218	**1218**	0.00
27	LA27	20 x 10	1235	1236	1235	1236	0.08
28	LA28	20 x 10	1216	1216	1216	1225	0.74
29	LA29	20 x 10	1157	1164	1164	1164	0.61
30	LA30	20 x 10	1355	1355	1355	**1355**	0.00
31	LA31	30 x 10	1784	1784	1784	**1784**	0.00
32	LA32	30 x 10	1850	1850	1850	**1850**	0.00
33	LA33	30 x 10	1719	1719	1719	**1719**	0.00
34	LA34	30 x 10	1721	1721	1721	**1721**	0.00
35	LA35	30 x 10	1888	1888	1888	**1888**	0.00
36	LA36	15 x 15	1268	1269	1268	1279	0.87
37	LA37	15 x 15	1397	1401	1397	1407	0.72
38	LA38	15 x 15	1196	1208	1196	1219	1.92
39	LA39	15 x 15	1233	1240	1233	1246	1.05
40	LA40	15 x 15	1222	1226	1224	1229	0.57

Notes:

BKS	Best known solution available in the literature
HPSO	Hybrid PSO algorithm (Xia and Wu, 2006)
PSO-VNS	Hybrid PSO with VNS (Tasgetiren et al., 2006)
D-PSO	Proposed Discrete PSO algorithm
RPI	Relative % increase in makespan over BKS

7. Conclusions

In this paper, a novel discrete version of particle swarm algorithm (PSO) is presented. The algorithm is tested using well-known bench-mark problem available in the literature. The solution obtained from the proposed algorithm is compared with best-known solution published in the literature. The performance of the algorithm is found to be good and able to achieve the best-known solution to 37 problems among 58 problems considered in this study. The success of this proposed algorithm is due to the novel solution construction procedure employed in this study. The weights assigned to the current, particle's best and global best particles is equivalent to the social and cognitive coefficients used in the conventional PSO algorithms used for solving the continuous function optimization problems. Further, the algorithm proposed in this study will be modified to suit the multi-objective optimization of the job-shops.

References

[1] Garey M R , and Johnson D S 1979 Computers and Intractability: a Guide to Theory of NP – Completeness *Freeman*. San Francisco.

[2] Jain A S, and Meeran S 1999 A state of the art review of job shop scheduling techniques *European Journal of Operational Research.* **113** 390-434.

[3] McKay K N, Safayeni F R, and Buzacott J A 1988 Job-shop scheduling theory: What is relevant? *Interfaces.* **18(4)** 84-90.

[4] Holthaus O, and Rajendran C 1997 New dispatching rules for scheduling in a job shop — An experimental study The *International Journal of Advanced Manufacturing Technology.* **13 (2)** 148–153.

[5] Yamada T and Nakano R 1997 Job shop scheduling *IEE control Engineering series.* 134-134.

[6] Błażewicz J, Domschke W, and Pesch, E., 1996 The job shop scheduling problem: Conventional and new solution techniques *European journal of operational research.* **93(1)** 1-33.

[7] Błażewicz J, Ecker K H, Pesch E, Schmidt G, and Weglarz J 2007 *Handbook on scheduling: from theory to applications.* Springer Science & Business Media.

[8] Jones A, Rabelo L C, and Sharawi AT 1999 Survey of job shop scheduling techniques *Wiley encyclopedia of electrical and electronics engineering.*

[9] Wang S F, and Zou Y R 2003 Techniques for the job shop scheduling problem: a survey *Systems Engineering-Theory & Practice.* **23(1)** 49-55.

[10] Thenarasu M, Rameshkumar K, and Marimuthu P (in press) Simulation Modeling and Development of Analytic Hierarchy Process (AHP) based Priority Dispatching Rule (PDR) for a Dynamic Press Shop *International Journal of Industrial and Systems Engineering.*

[11] Kennedy J, and Eberhart R C 1995 Particle swarm optimization *Proceedings of IEEE International Conference on Neural Networks.* Piscataway, NJ, USA, 1942-1948.

[12] Clerc M 2004 Discrete particle swarm optimization, illustrated by the Traveling Salesman Problem *New Optimization Techniques in Engineering.* Heidelberg Germany Springer 219-239.

[13] Tasgetiren F M, Liang Y-C, Sevkli M, and Gencyilmaz G 2007 Particle Swarm optimization for makespan and Total flowtime minimization in permutation flowshop sequencing problem *European Journal of Operational Research.* **177** 1930–1947.

[14] Tasgetiren M F, Sevkli M, Liang Y-C, and Yenisey M M 2006. A particle swarm optimization and differential evolution algorithms for job shop scheduling problem *International Journal of Operations Research.* **3(2)** 120-135.

[15] Xia W J, and Wu Z M 2006 A hybrid particle swarm optimization approach for the job-shop scheduling problem *The International Journal of Advanced Manufacturing Technology.* **29(3)** 360-366.

[16] Rameshkumar K, Suresh R, and Mohanasundaram K 2005 Discrete particle swarm optimization (DPSO) algorithm for permutation flowshop scheduling to minimize makespan *LNCS - Advances in Natural Computation.* **3612** 572-581.

[17] Rameshkumar K Rajendran C Mohanasundaram K M 2011 Discrete particle swarm optimisation algorithms for minimising the completion-time variance of jobs in flowshops *International Journal of Industrial and Systems Engineering.* **7(3)** pp.317-340.

[18] Rameshkumar K, Rajendran C, and Mohanasundaram K M 2012. A novel particle swarm optimisation algorithm for continuous function optimisation *International Journal of Operational Research.* **13(1)** 1-21.

[19] Karthi R, Rajendran C, and Rameshkumar K 2011 Neighborhood Search Assisted Particle Swarm Optimization (NPSO) Algorithm for Partitional Data Clustering Problems *Advances in Computing and Communications.* 552-561.

[20] Kadadevaramath R S, Chen J C, Shankar B L, and Rameshkumar K 2012 Application of particle swarm intelligence algorithms in supply chain network architecture optimization *Expert Systems with Applications.* **39(11)** 10160-10176.

[21] Sun Y, and Xiong H 2012 Job-shop scheduling problem based on particle swarm optimization algorithm *Sensors & Transducers.* **16** 116.

[22] Gao H, Kwong S, Fan B, and Wang R 2014 A hybrid particle-swarm tabu search algorithm for solving job shop scheduling problems *IEEE Transactions on Industrial Informatics.* **10(4)** 2044-2054.

[23] Meng Q, Zhang L, and Fan Y 2016 A Hybrid Particle Swarm Optimization Algorithm for Solving Job Shop Scheduling Problems *Asian Simulation Conference.* 71-78 Springer Singapore.

[24] Gen M, Tsujimura Y, and Kubota E 1994 Solving job-shop scheduling problems by genetic algorithm *Systems, Man, and Cybernetics.* **2** 1577-1582.

[25] French H 1982 *Sequencing and Scheduling – An introduction to the mathematics of the job shop.* Ellis Hoewood John Wiley & Sons New York.
[26] Nakano R, and Yamada T 1991 Conventional genetic algorithm for job shop problems *ICGA.* **91** 474-479.
[27] Lawrence S 1984 Resource constrained project scheduling: An experimental investigation of heuristic scheduling techniques *GSIA.* Carnegie Mellon University Pittsburgh PA.
[28] Adams J, Balas E, and Zawack D 1988 The Shifting bottleneck procedure for job shop scheduling *Management Science.* **34**, 391-401.
[29] Fisher H, and Thompson G L 1960 Probabilistic learning combinations of local job shop scheduling rules Muth J F Thompson GL (Eds.) *Industrial scheduling.* Prentice Hall Englewood Cliffs NJ 225-251.
[30] Applegate D, and Cook W 1991 A computational study of the job shop scheduling problem, *ORSA Journal on Computing.* **3** 149-156.