REVISTA INVESTIGACIÓN OPERACIONAL VOL. 42, NO. 2, 195-203, 2021

IMPROVED PARTICLE SWARM ALGORITHM FOR PERMUTATION FLOW SHOP SCHEDULING PROBLEMS

Manal Abdulkareem Zeidan*, Manar Abdulkareem Al-AbajI and Manaf Hazim Ahmed
University of Mosul, Irak.

ABSTRACT

Particle Swarm Optimization PSO is one of the appropriate methods for solving NP-hard problems. So, improving PSO has sense. The permutation flow shop scheduling is one of these problems. In this paper, the permutation flow shop scheduling problems is solved by using improved particle swarm optimization named IPSO. The improvement is done by replacing the generated initial swarm by another one near to optimal solution by implementing the Individual improvement scheme IIS. The performance of IPSO is evaluated by several evaluation criteria and several problems generated randomly from uniform distribution and compered with PSO.The result shows that IPSO is outperform PSO.

KEYWORDS: Particle swarm optimization, Flow shop scheduling problems, Individual improvement scheme

MSC: 90C59

RESUMEN

La optimización por enjambre de partículas PSO es uno de los métodos apropiados para resolver problemas NP-duro. Por lo tanto, es importante proponer mejoras al método PSO. En este articulo, se considera el problema NP-duro correspondiente al modelo de flujo permutacional. Se propone una mejora al PSO que llamamos IPSO. La misma consiste en sustituir el enjambre inicial generado por otro cercano a la solución óptima mediante la implementación del esquema de mejora individual IIS. del enfoque PSO q. El rendimiento de IPSO se mide mediante varios criterios de evaluación y varios problemas generados aleatoriamente a partir de una distribución uniforme y comparamos los resultados con los de la PSO. Los resultados muestran que IPSO supera a PSO.

PALABRAS CLAVE: Optimización por enjambre de partículas, problema de horario de flujo permutacional, esquema de mejoramiento individual

1. INTRODUCTION

It is well known that the scheduling problems have many applications related to all domains. One of its goals is to reduce execution time for a mission. For example, implanting scheduling concepts in order to mange the cranes in sea ports, so that the ships are emptied from the containers in minimum time [18]. These scheduling problems have many environments, each of them treats a specific system. Flow shop problem is one of common scheduling environmens. It deals with processing jobs on a several machines in order to attain specific objective e.g (minimize makespan, maximum lateness, minimize sum of flow times .... etc) . The permutation flow shop scheduling problem PFFSP is a flow

*manal.alabaji@uomosul.edu.iq

shop problem when the jobs will be processed in the same order in each machine. It is stated as a one of difficult and complex optimization problems which is classified as NP-hard [9]. The first algorithm which deals with this problem for two and three machines is introduced by [16]. It is well known as Johnson algorithm. Due to the computational complexity of this problem, a several metaheuristics methods have been introduced and many improvements on these methods have been done.

The researchers in [11] have included one of the local search methods into PSO for minimizing the makespan of PFSSP. The method which has been used is very easy and efficient which is called Variable Neighbourhood Search VNS. Whereas the researchers in [12] have generated one of the particles in the initial swarm in PSO by Nawaz-Enscore-Ham NEH heuristic to solve a flow shop problem. An efficient method to solve large flow shop problems is introduced by [13], by combining the genetic operator (mutation) with PSO. An efficient algorithm called Hybrid Particle Swarm HPSO has been introduced in [14]. It combines both the Knowledge Evolution Algorithm KEA with PSO to give best sequence so that minimize the makepsan. In [15] a proposed algorithm called MPSOMA is based on a memetic algorithm. They adopted a procedure that divides the particle swarm population into three sub-populations. In each sub-population there is a particle that develops itself by classical PSO. And then, Variable Neighbourhood Search VNS and Individual Improvement Scheme IIS local search methods have been used.

In this paper, Improved version of Particle swarm optimization has been presented, named improved Particle swarm optimization IPSO. The improvement which has been done in PSO is more efficient and stable in finding the best sequence which gives minimum makespan. The local search method called Individual Improvement Scheme IIS has been used in the initial particle swarm in order to raise the performance of PSO. In another word, using IIS in initial particle swarm for starting with near optimal solution leads to get better outputs. To make PSO applicable to PFFSP, a Ranked-Order-Value ROV has been used.

In order IPSO to be reliable in finding best sequence of jobs, its quality is assessed by several criteria compared with classical PSO. These criteria are Best Percentage Relative Deviation BPRD from the lower bound, Average of Percentage Relative Deviation APRD, Ratio of Best (respectively, Worst) Percentage Relative Deviation RBPRD (receptively, RWPRD). Furthermore, the stability of the algorithm has been investigated. The two algorithms, IPSO and PSO have been implemented on several problems generated randomly with uniform distribution and selecting different jobs and machines. The criteria results show that IPSO is more effective, stable and reliable than PSO in all problems.

This paper is organized as follows: The description of PFSSP and its details are presented in Section 2. Section 3. is devoted to describe the Particle swarm optimization and its algorithm. The improvement of PSO and its evaluation with the results are illustrated in section 4.. Finally, the conclusion is discussed in 5.

2. PROBLEM DESCRIPTION

The permutation flow shop scheduling PFSSP aims at finding the best arrangement of scheduling n jobs through m machine that minimize makespan. In this problem the job i, $i = 1, ..., n$ is proceed at machine j, $j = 1, ..., m$ with processing time $T_{i,j}$. The job i includes m operations, each operation proceeds on specific machine. All jobs will proceed through the machines in the same order, i.e. $J_i = \{O_{i1}, O_{i2}, ..., O_{im}\}$ and each machine processes the jobs in the same sequence. Mathematically, PFSSP is expressed by $n/m/F/C_{max}$ where n is the total number of jobs, m is the total number of machines, F refers to flow shop problem and C_{max} is the makespan. The permutation job set is denoted by $\pi = (\pi_1, \pi_2, ..., \pi_n)$. The completion time of processing job π_j at machine m is denoted by $C(\pi_i, m)$. The makespan of PFSSP depending one the permutation job shop set π can be calculated as follows:

$$\begin{aligned}
C(\pi_1, 1) =& T_{\pi_1,1}\\
C(\pi_i, 1) =& C(\pi_{i-1}, 1) + T_{\pi_i,1}, \quad i = 2, ..., n\\
C(\pi_1, j) =& C(\pi_1, j-1) + T_{\pi_1,j}, \quad j = 2, ..., m\\
C(\pi_i, j) =& \max\{C(\pi_{i-1}, j), C(\pi_i, j-1)\} + T_{\pi_{i,j}} \qquad i = 2, ..., n, j = 2, ..., m
\end{aligned}$$

Then the makespan is

$$C_{max}(\pi) = C(\pi_n, m).$$

For more information see [1] and [2].

3. PARTICLE SWARM OPTIMIZATION PSO

Particle swarm optimization PSO is a Metaheuristic method proposed by [17]. It is an algorithm inspired by the collective behaviour of the bird swarm in searching the food. This algorithm starts with a group of particles called swarm generated randomly. These particles move in d-dimensional searches space to get better position.

Each iteration a particle i in the swarm regulate it's velocity according to it's best position P_{best}; the best position in swarm G_{best} and the previous velocity. After that, the particle i updates its position according to recently created velocity and its last position. The following equations are used to update the velocity and the position of particle:

$$V_i^t = W * V_i^{t-1} + C_1 * Rand_1 * (P_{best_i}^{t-1} - X_i^{t-1}) + C_2 * Rand_2 * (G_{best}^{t-1} - X_i^{t-1}) \tag{3.1}$$

$$X_i^t = X_i^{t-1} + V_i^t, \tag{3.2}$$

where V_i and X_i receptively are the current velocity and position of particle, W is inertia weight, C_1 is the learning factor, C_2 is the social learning factor and the random numbers $Rand_1, Rand_2$ generated from uniform distribution.

The standard PSO is designed to solve the problems in continuous search space, while the search space for PFSSP is discrete. So, PSO is not appropriate to be applied directly for solving PFSSP. Thus, the first matter should be done in order to make PSO applicable for PFSSP is to find a different solution representation suitable for PFSSP; the second matter is to find an approach for converting the continuous values for position to a discrete permutation of job sequence. For more information see [3], [4], [5] and [6].

Usually, for PFSSP the solution representation is a string of permutation for n jobs indicate by $(1, 2, ..., n)$, see [6]. For example the solution representation of scheduling 6 jobs at m machines is as below, indicating that the sequence of processing the jobs at each machine is Job 3 $\Rightarrow$ Job2 $\Rightarrow$ Job1 $\Rightarrow$ Job6 $\Rightarrow$ Job5 $\Rightarrow$ Job4 .

3	2	1	6	5	4

The Ranked-Order-Value ROV rule presented in [7] is a heuristic approach widely used to convert the continuous values for position to a discrete permutation of job sequence. It works as follows: ranking

Position	1	2	3	4	5
Position value	1.43	-3.52	0.51	-1.78	3.27

of value

-3.52	-1.78	0.51	1.43	2.27
2	4	3	1	5

Algorithm (1) PSO for PFSSP

1. Generate the initial population for N particles.
2. Evaluate the fitness (mackespan) for each particle in the population.
3. Find $P_{best,i}$ for each particle and G_{best} for all particles.
4. **While** a stop condition (the optimal solution is found or maximal number of iterations is reached) is not achieved **do**.
5. **for** each particle i **do**.
6. By Eq 3.1 update the velocity of each particle.
7. By Eq 3.2 update the position of each particle.
8. Apply the ROV rule for converting the continuous position values to permutation job sequence.
9. Evaluate the fitness of the new particle position.
10. Update the P_{best} for each particle and the G_{best} for all particle.
11. **end for**
12. **end while**

4. IMPROVED PSO ALGORITHM IPSO

The effectiveness of the metaheuristics methods will be better if it starts with good solution in the initial population. For this reason, an improvement in the practical swarm optimization has been done in this section by applying one of the local search method called Individual Improvement Scheme IIS [15] to each particle which is generated randomly. The implementation of IIS method before starting with step of PSO will increase its performance. It works as in the following steps:

Algorithm (2) Individual Improvement Scheme IIS

1. **for** each job i **do**.
2. **for** each job in the partical after i **do**.
3. Exchange the position of job with position of job after itself.
4. Calculate the makespan of the new particle.
5. Update the particle according to an improvement in makespan.
6. End **for**.
7. End **for**.

First of all, in order to validate IPSO, the algorithm PSO will be the control for all examinations. The reasons behind choosing PSO as control are: IPSO is derived from the spirit of PSO and it is scientifically supported in many researches. So, all the criteria that will apply to IPSO will be applied to PSO also for all problems which are generated.

The algorithm IPSO is implemented independently for $K = 20$ times on each generated problem. Two criteria are implemented to assess the output quality of IPSO compared with PSO. The first one, is defined as the Best Percentage Relative Deviation $BPRD$ of the makepsan C_{max} from the lower bound LB. Mathematically

$$PRD_k = \frac{C_{max,k} - LB}{LB} \times 100, \tag{4.1}$$

such that,

$$BPRD = \min_k \{PRD_k\}, \forall k = 1, 2, ..., K, \tag{4.2}$$

Furthermore, the Average Percentage Relative Deviation APRD is,

$$APRD = \frac{1}{K} \sum_{\forall k=1,...,K} PRD_k. \tag{4.3}$$

Although, BPRD and APRD are trustworthy in evaluating the performance of IPSO, this evaluation captured by these criteria will be general. For more precise evaluation, the second criteria has been implemented. It is concerned with the ration of the best and worst C_{max} obtained by IPSO to the number of generation. This criteria will focus on the ability of this algorithm in arranging the jobs, such that which gives nearest C_{max} to the lower bound. The ratio of the risk is that the algorithm falls in the wrong arrangement of jobs, so that C_{max} stray from the lower bound is significant also.
The evaluation mechanism in this method depends on the u-quantile of the best and worst Percentage Relative Deviation PRD of C_{max} has been obtained by the algorithm. The u-quintile will be calculated according to PRD obtained by PSO. Of course, the best C_{max} has minimum PRD as well as the worst C_{max} has the maximum. So, evaluating the best and worst C_{max} or the corresponding PRD is equivalent. The consideration in this paper that all $C_{max,k}$ have PRD_k resulted from IPSO less than or equal to the 0.1-quantile from all the PRD resulted from PSO are the best C_{max} and the worst will be all $C_{max,k}$ have PRD_k more than or equal to $0.9-$ quantile.
Let $Y := PRD_{IPSO}$ (receptively, $X := PRD_{PSO}$) be the percentage relative deviation resulted from IPSO (respectively, PSO). And let $q(u, \cdot)$ be the $u-$quantile of $(\cdot)$. A subset $Y^B := \{Y_k : Y_k \leq q(0.1, X)\}$ is the best PRD in the set Y and the worst PRD defined as the following $Y^W := \{Y_k : Y_k \geq q(0.9, X)\}$, such that $Y^B, Y^W \subset Y$. Then the Ratio of the Best Percentage Relative Deviation RBPRD is

$$RBPRD = \frac{\#Y^B}{K}, \tag{4.4}$$

and the Ratio of the Worst Percentage Relative Deviation RWPRD is

$$RWPRD = \frac{\#Y^W}{K}. \tag{4.5}$$

As we mentioned above, IPSO validation is confirmed by assessing the performance of it compared with PSO. These two algorithms are applied to several flow shop problems. The processing time of each job is generated randomly from discrete uniform distribution with range $[1, 20]$. Matlab (R 2015 a) on laptop with intel(R) core i5, 2.40 GHZ processor and 4 GB memory has been used in coding the two algorithms.The results graphs designed in R. The IPSO and PSO are parameterized as follows: swarm-size=100, $C_1 = C_2 = 2$, $W = 0.9$ and 300 iteration.The numerical results of the lower bound LB, makespan C_{max} and Best Percentage Relative Deviation BPRD of each PSO and IPSO are shown in the Table 1.
The $BPRD$ in the Table 1 ensures, that IPSO is outperform PSO for all generated problems except the problem with 20 machines and 10 jobs, the performance was equivalent. In another word, IPSO overcome on PSO in finding the best sequence of jobs gives minimum C_{max}. It is worth mentioning that the stability of algorithm in finding the C_{max} provides confidence in the algorithm outputs. Figure

Table 1: The lower bounds LB, the makespans C_{max} and the Best Percentage Relative Deviation $BPRD$ of each PSO and IPSO for all generated problems

			PSO		IPSO	
No. of machines	No. of jobs	LB	C_{max}	$BPRD$	C_{max}	$BPRD$
5	10	123	125	1.63	124	0.81
	20	259	268	3.47	263	1.54
	50	574	601	4.70	592	3.14
	100	1215	1226	0.91	1218	0.25
10	10	206	234	13.59	231	12.14
	20	317	348	9.78	340	7.26
	50	597	669	12.06	650	8.88
	100	1094	1247	13.99	1184	8.23
15	10	237	287	21.10	286	20.68
	20	358	442	23.46	431	20.39
	50	695	806	15.97	771	10.94
	100	1248	1429	14.50	1378	10.42
20	10	277	323	16.61	323	16.61
	20	432	519	20.14	515	19.21
	50	754	906	20.16	869	15.25
	100	1251	1500	19.90	1429	14.23

1 illustrates, the stability of IPSO compared with PSO for all the problems generated, furthermore APRD .

Figure 1 shows that, although the stability in finding the best solution has been investigated for the two algorithms, IPSO is more stable. This additional stability is due to implementing IIS algorithm on initial population, such that this algorithm brings the particles to search for best sequence of jobs near in the optimal solution space. It is to be noted that, IPSO in problem with 5 machines and 100 jobs gives approximately exact solution for all iterations. In other words, 100% is stable in solution. Also IPSO outperform on PSO by $APRD$ for all problems.

For compactness on the outputs of IPSO algorithm, RBPRD and RWPRD criteria have been applied and it results are illustrated in the Figure 2.

Form all problems when the number of jobs is small, RBPRD of IPSO and PSO are approximately equivalent. This is due to the number of possibilities of the jobs arrangement are not relatively high. For that reason, the two algorithms can reaches to the optimal sequence which gives minimum C_{max}. But RBPRD of IPSO is very high and sometimes reaches to one when the number of jobs high. This indicates that IPSO is very capable and reliable algorithm can be depended on in the arrangement of the jobs to get minimal makespan. While, PSO's performance decreases with increasing the jobs.

RWPRD in Figure 2 indicates the risk of algorithms in choosing arrangement of jobs have C_{max} far from the lower bound. In all problems,RWPRD of C_{max} obtained by IPSO are very low, sometimes reach to zero. While in PSO are very high. That means keeping C_{max} near form the lower bound is very high when implementing IPSO. In another word, IPSO is more reliable than PSO.

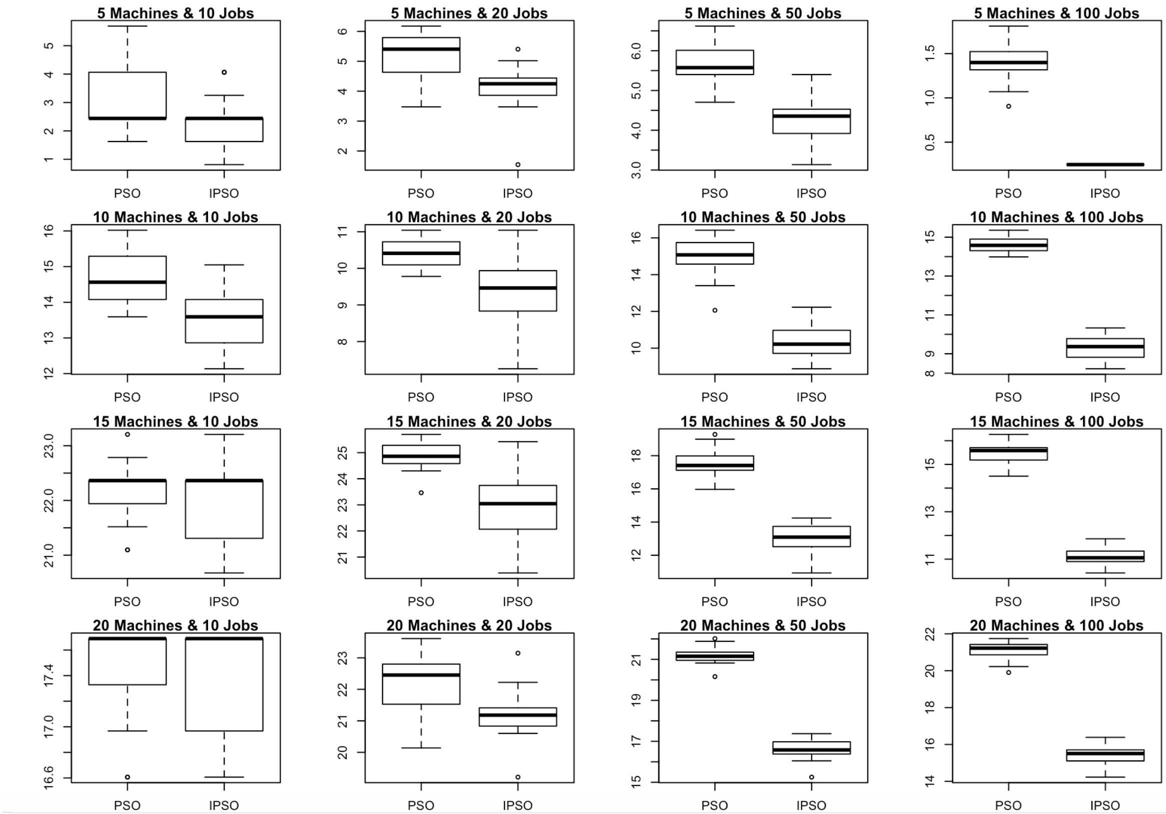

Figure 1: The boxplots represents the Average Percentage Relative Deviation $APRD$ and its variation obtained by PSO and IPSO for all generated problems.

5. CONCLUSION

An improvement particle swarm algorithm IPSO has been proposed for solving NP-hard permutation flow shop scheduling problems PFSSP. The improvement has focused on the initial swarm in particle swarm algorithm PSO. The enhancement is done by implementing Individual Improvement Scheme IIS local search method on each particle in the swarm. The IPSO performance has been tested by several criteria and several problems compared with classical PSO. All the criteria methods BPRD, APRD, RBPRD and RWPRD are implemented in the evaluation as well as the stability of the algorithm. All the results show that, IPSO is more efficient and reliable algorithm than PSO . It is very recommended in the arrangement that such jobs give the optimal makespan, especially in big problems.

ACKNOWLEDGEMENT

The authors are very grateful to the University of Mosul/ College of Computer science and Mathematics and College of Education for Pure Science for their help and facilities, which helped to improve the quality of this work.

RECEIVED: DECEMBER, 2019.
REVISED: JUNE, 2020.

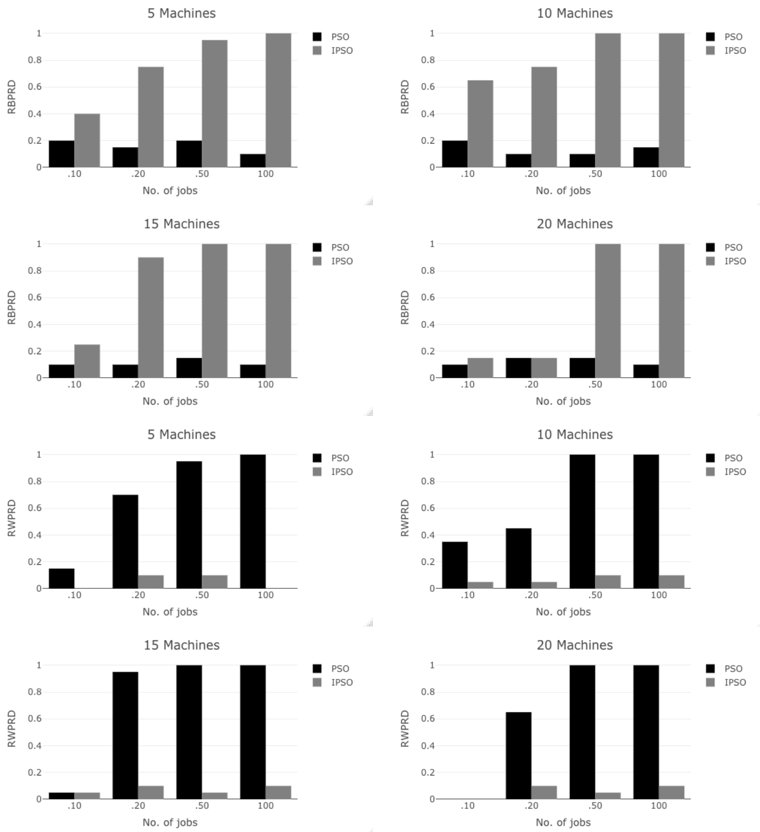

Figure 2: The barplots represents the Ratio of the Best Relative Deviations RBPRD of C_{max} form the lower bound as well as the ratio of the Worst Relative Deviations RWPRD of C_{max}.

REFERENCES

[1] LI X. and M. YIN, (2013)¿A hybrid cuckoo search via Lévy flights for the permutation flow shop scheduling problem, **International Journal of Production Research**, 51, 4732–4754 .

[2] AKHSHABI,M., J. HADDADNIA and M. AKHSHABI (2012)., Solving flow shop scheduling problem using a parallel genetic algorithm, **Procedia Technology**, 1, 351–355

[3] LIN, T.L., S.J. HORNG, T.W. KAO, Y.H. CHEN, R.S. RUN, R.J. CHEN, J.L. LAI and I.H. KUO (2010): An efficient job-shop scheduling algorithm based on particle swarm optimization, **Expert Systems with Applications**, 37, 2629–2636

[4] SANTOSA,B.., N. SISWANTO and others,(2018): Discrete particle swarm optimization to solve multi-objective limited-wait hybrid flow shop scheduling problem, **IOP Conference Series: Materials Science and Engineering**, 337, 012006

[5] HAJINEJAD, D., N. SALMASI and R. MOKHTARI,(2011): A fast hybrid particle swarm optimization algorithm for flow shop sequence dependent group scheduling problem, **Scientia Iranica**, 18 759–764

[6] BEWOOR L.A., V.C. PRAKASH and S.U. SAPKAL,(2018): Production scheduling optimization in foundry using hybrid Particle Swarm Optimization algorithm, **Procedia Manufacturing**, 22, 57–64

[7] BEAN, J.C.,(1994): **Genetic algorithms and random keys for sequencing and optimization**, ORSA journal on computing, 6, 154–160

[8] ALDOWAISAN T. and Q. ALLAHVERDI,(2003): New heuristics for no-wait flowshops to minimize makespan, **Computers & Operations Research**, 30, 1219–1231

[9] ZHANG, S.,(2010): Large-scale flow shop scheduling based on genetic algorithm, **Education Technology and Computer (ICETC), 2nd International Conference**, 1, V1–308

[10] RAMANAN,T.R., M. IQBAL and K. UMARALI,(2014): A particle swarm optimization approach for permutation flow shop scheduling problem, **International Journal for Simulation and Multidisciplinary Design Optimization**, 5, A20

[11] TASGETIREN, M.F. ,M. SEVKLI, T.C. LIANG and G. GENCYILMAZ,(2004): Particle swarm optimization algorithm for permutation flowshop sequencing problem, **International Workshop on Ant Colony Optimization and Swarm Intelligence**, 382–389

[12] CHANDRASEKARAN, S., S.G. PONNAMBALAM, R.K. SURESH and N. VIJAYAKUMAR,(2006): A hybrid discrete particle swarm optimization algorithm to solve flow shop scheduling problems, **Cybernetics and Intelligent Systems, IEEE Conference**, 1–6

[13] ZHANG, C., J. SUN, X. ZHU and Q. YANG,(2008): An improved particle swarm optimization algorithm for flowshop scheduling problem, **Information Processing Letters**, 108, 204–209

[14] TANG, H. and C.M. YE,(2010): Permutation flow shop scheduling algorithm based on a hybrid particle swarm optimization, **Industrial Engineering and Engineering Management (IE&EM), IEEE 17Th International Conference** , 557–560

[15] LIU, R., C. MA, W. MA and Y. LI,(2013): A multipopulation PSO based memetic algorithm for permutation flow shop scheduling, **The Scientific World Journal**, 2013 11

[16] JOHNSON, S.M.,(1954): Optimal two-and three-stage production schedules with setup times included, **Naval research logistics quarterly**, 1, 61–68

[17] KENNEDY J. and R. EBERHART,(1995): Particle swarm optimization, **Proceedings of IEEE International Conference on Neural Networks IV**, pages, 1000

[18] ALSOUFI,G., X. YANG and A. SALHI,(2018): Combined quay crane assignment and quay crane scheduling with crane inter-vessel movement and non-interference constraints, **Journal of the Operational Research Society**, 69, 372-383