VOL. 11, NO. 12, JUNE 2016 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

COMBINED-OBJECTIVE OPTIMIZATION IN IDENTICAL PARALLEL MACHINE SCHEDULING PROBLEM USING PSO

Bathrinath S.[1], Saravanasankar S.[1] and Ponnambalam SG.[2]
[1]Department of Mechanical Engineering, Kalasalingam University, Krishnankoil, Tamilnadu, India
[2]School of Engineering, Monash University Malaysia, Bandar Sunway, Selangor, Malaysia
E-Mail: bathri@gmail.com

ABSTRACT

Identical Parallel Machine Scheduling (IPMS) problem for minimizing make span and number of tardy jobs simultaneously is considered as very important production scheduling problem but there have been many difficulties in solving large scale IPMS problem with too many jobs and machines. In order to minimize make span and number of tardy jobs simultaneously improved versions Particle Swarm Optimization (PSO) is proposed to enhance scheduling efficiency with less computational burden. The premature convergence at the initial stages of iteration is considered as the major drawback for standard PSO. However, this can be avoided by incorporating mutation a common genetic algorithm operator into the standard PSO and is termed as MPSO. Several numerical examples demonstrate the MPSO proposed is efficient and fit for large scale IPMS problem for minimizing the objectives considered. The solution obtained by MPSO outperforms standard PSO.

Keywords: make span, tardy jobs, parallel machines, PSO, mutation.

1. INTRODUCTION

Identical parallel machine scheduling problem for minimizing make span and minimizing number of tardy jobs simultaneously are proved to be NP-Hard problems [1,2]. Traditionally these problems can be solved by operational methods such as branch and bound method, dynamic programming, integer programming [3-8] etc. Heuristic methods are suitable for small size problems [9] but in case of large size problems methods like LPT, MULTIFIT, LISTFIT etc. are not able to produce effective solution when the accuracy of the solution needs to be improved. Particle Swarm Optimization (PSO) [10-13 and 17] has been applied for its characteristics such as easy realization, high speed and its robustness in case of combinatorial optimization problems. Moreover, it has also shown great advantages in solving industrial production scheduling problems. The scheduling in manufacturing industries received much attention by the researchers to deliver the products to the customers in terms of quality and quantity in time. In this article, the authors propose a novel MPSO method for solving IPMS problems. The computational results were compared with standard PSO. It shows that MPSO proposed in this paper is suitable for solving large size IPMS problems for minimizing make span and minimizing number of tardy jobs simultaneously.

In Section 2, the problem is mathematically formulated. The detailed optimization algorithm of the problem is explained in Sections 3. The computational results are discussed in Section 4. Finally conclusions are discussed in Section 5.

2. PROBLEM DESCRIPTION

Let $N = \{J_1,J_2,...J_i...,J_n\}$ of n jobs are to be scheduled on a set $M = \{M_1,M_2,...M_j...,M_m\}$ of m identical parallel machines. The first objective of this IPMS problem is to find the optimal schedule $S = \{S_1,S_2,...S_j...,S_m\}$ where S_j is a subset of jobs assigned to machine M_j, such that $\max\{C_1(S), C_2(S),...C_j(S)...,C_m(S)\} = C_{max}(S)$ is minimum where,

$$C_j(S) = \sum_{p_i \text{ of } J_i \in S_j} p_i \tag{1}$$

The second objective is to minimize the number of tardy jobs

$$N_t = \sum_{j=1}^{n} U_j \tag{2}$$

where U_i will assume the value 1 for the job J_i if the job J_i is completed in time, that is, if the due date d_i for the job J_i is smaller than completion time C_i otherwise, 0.

The combined objective function (COF) is formulated as follows and is given in equation (3)

MinCOF=

$$\delta * \frac{\text{Makespan}}{\text{Sum of processingtimes}} + (1-\delta) * \frac{\text{Number of tardyjobs}}{\text{totalnumber of jobs}} \tag{3}$$

where δ is the weightage factor assigned and its value is $0 < \delta < 1$. The assumptions to be considered for solving IPMS problems are as follows:

Assumption:

a) All jobs can be processed on any of the parallel machines.
b) Each of the parallel machines can process at most one job at a time.
c) Each job has to be processed without interruption on one of the machines.
d) No job can be processed by more than one machine.
e) All jobs are available to be processed at time zero.
f) No down time is considered, and set up time is included in the processing time.

3. PROPOSED PSO BASED HEURISTIC FOR IPMS PROBLEMS

The PSO was first introduced by Kennedy and Eberhart [10] in 1995 which was inspired by social behavior of bird flocking and fish schooling, also a population based bio-inspired optimization algorithm. In PSO, particles are termed as potential solutions; each particle adjusts its position based on its own experience as well as the experience of a neighboring particle. It has a quality of adapting the global and local exploration capabilities. PSO has always maintained its flexibility towards reaching the global optimal solutions. The particles in PSO have their own memory. It enables to retain good solutions among all particles. The previous best value is called as pbest of the particle and gbest is called as the best value of all the particles among the pbest in the swarm. The PSO has been repeated by the following steps after evaluation. (i) pbest is to be updated if a better value is discovered. (ii) The velocities of all the particles are updated based on the experiences of pbest and gbest in order to update the position of each particle with the velocities currently updated. (iii) Permutation is to be determined so that the evaluation is again performed to compute the fitness.

After finding pbest and gbest, the velocities and positions of each particle are updated using the equation (3.1) and (3.2) respectively.

$$v_{ij}^t = w^{t-1} v_{ij}^{t-1} + c_1 r_1 \left(p_{ij}^{t-1} - x_{ij}^{t-1} \right) + c_2 r_2 \left(g_j^{t-1} - x_{ij}^{t-1} \right) \tag{4}$$

$$x_{ij}^t = x_{ij}^{t-1} + v_{ij}^t \tag{5}$$

where v_{ij}^t represents the velocity of the particle i at iteration t with respect to j^{th} dimension where j = 1,2,..,n. c_1 and c_2 are social and cognitive parameters and r_1 and r_2 are uniform random numbers between (0,1).

$$w = w_{t-1} * \beta \tag{6}$$

where β is decrement factor.

a) Solution representation

The solution representation is considered as the major issue in designing PSO. In general, the particle does not have the capability of doing permutation by itself. Tasgetiren *et al*. [14] introduced SPV rule to determine the permutation implied by the position values x_{ij}^t as shown in Table-1.

Table-1. SPV rule.

Dimension, j	1	2	3	4	5	6
x_{ij}^t	2.13	1.21	**-3.21**	-2.12	0.97	-1.91
v_{ij}^t	1.64	-0.32	-1.24	1.54	-0.79	0.62
Jobs, Π_{ij}^t	**3**	4	6	5	2	1

The least position in the table is x_{i3}^t = -2.92. So the dimension j = 3 is assigned to the first job Π_{i1}^t = 3 in the permutation Π_i^t. Similarly, next least values are arranged in sequence. The different permutations at each iteration t can be obtained by updating the position of each particle. The initial sequence of jobs is $\Pi_{ij}^t = [3,4,6,5,2,1]$. Once the sequence of jobs have been calculated by SPV rule, the jobs have been assigned to the machines with the same sequence. Let p_j be the processing time of n number of jobs as shown in Table-2.

Table-2. Processing times for the jobs.

Jobs j	1	2	3	4	5	6
p_j	6	8	10	9	12	11

Let there be two machines (m=2) and six jobs (n=6). According to the SPV rule, third job occupies first position in the initial sequence. The corresponding processing time of the particular job has been taken from the Table-2 and placed in the first machine as shown in Table-3. Similarly, fourth job in the initial sequence occupies second position. So the corresponding processing time for the fourth job from Table-3 has been placed in the second machine. Two different jobs have been assigned to two different machines. From these machines, the machine which completes the assigned job first is assigned with a new job from the sequence in Table-1. In this case, the second machine assigned with job number 4 completes the job first as the processing time for the job is seven units of time. The third position in the sequence is job six having the processing time of eleven units of time. Hence, the second machine is occupied by the job number 6. This total process is repeated till all the jobs are assigned to machines.

Table-3. Assignment of jobs using SPV rule.

Machines m_i	1	2

p_{ij}	10(3)
	12(5)
	6(1)
		9(4)
		11(6)
		12(2)

From the Table-3, it is clear that job completion time of first machine is 28 units of time and that of second machine is 32 units of time. Therefore, the makespan has been considered as the maximum of the completion time in all the machines. In this case, makespan is considered as 32 units of time. From the Table-3, p_{ij} denotes the processing time of the j^{th} jobs in i^{th} machine where 10(3) represents third job is processed having 10 units of time.

b) Mutation operator

Mutation is an important operator of the genetic search as it helps to avoid local optimal by preventing the population of chromosomes [18]. While considering PSO, the lack of population diversity among the swarms is known as a factor for the convergence on local minima. Hence, the PSO has been incorporated with mutation operator to enhance its global search capacity and to improve the performance characteristics. In this paper, a non-uniform mutation operator proposed by Michalewicz's [15] has been incorporated in PSO. If the number of iterations with no diversity in the solution exceeds, mutation operation is carried out. This operator works by changing a particle position dimension.

$$Mutate(x_{iD}) = \begin{cases} x_{iD} + delta\ (t,\ U - x_{iD}) : rb = 1 \\ x_{iD} + delta\ (t,\ x_{iD} - L) : rb = 0 \end{cases} \qquad (7)$$

where t is the current iteration number, U is the upper bound value of the particle dimension, L is the lower bound value of the particle dimension, rb is the randomly generated bit and delta (t,y) returns the value in the range $[0:y]$.

$$delta\ (t,y) = y\left(1 - r^{(1-t/T)^{b}}\right) \qquad (8)$$

where r is the random number generated from the uniform distribution range [0:1], T is the maximum number of iteration, b is the tunable parameter which is set as 5. The pseudo-code for the proposed MPSO is given as follows:

c) Computational experiments

The study aims to analyze the performance of all the developed intelligent search techniques to minimize makespan and number of tardy jobs simultaneously using COF for the identical parallel processor scheduling problem considered in this chapter. The algorithms are coded in MATLAB R2010a and executed in Intel® Core ™ i5 CPU M430 @ 2.27 GHz with 4GB RAM. The benchmark problems have been taken from Tanaka *et al*. [8] and also available in (https://sites.google.com/site/shunjitanaka/pmtt). Fisher [16] has proposed the generic method for generating these problems. The integer processing times p_j $(1 \leq j \leq n)$ of these problems have been generated from the uniform distribution between [1,100]. The total processing times have been calculated by $P = \sum_{j=1}^{n} p_j$, the due dates d_j $(1 \leq j \leq n)$ have been generated using the uniform distribution $[P(1-\tau-R/2)/m, P(1-\tau+R/2)/m]$ where n is the number of jobs, m is the number of machines, τ is the tardiness factor and due date ranges are changed by n = 20, m={2,3,4,5,6,7,8,9,10}, τ = {0.2,0.4,0.6,0.8,1.0} and R = {0.2,0.4,0.6,0.8,1.0}.

For all combination of m, n, τ, R, five problems have been generated. There are five characteristics which are used to represent the problem, they are number of jobs, number of machines, the tardiness, the due date range and the position of the instance respectively. For example, the notation of problem 20_03_04_06_002 represents the first problem of three machines, 20-job, τ = 0.4, and R = 6. Among these benchmark problems, fifty problems have been randomly selected for testing and analysis of proposed algorithms. The weight values of δ are considered as 0.5 for calculating COF. MPSO parameter settings are as follows: the inertial weight w is set as 0.6, both social and cognitive parameters are set as 1.04 and the decrement factor β is set as 0.7. The maximum number of iterations taken for MPSO is 15000. When considering MPSO with PSO, it produces better results by 97.5% (Table-4). The Figures-1 and 2 represents Convergence curves for COF with PSO and MPSO respectively for the problem 20_10_02_02_004.

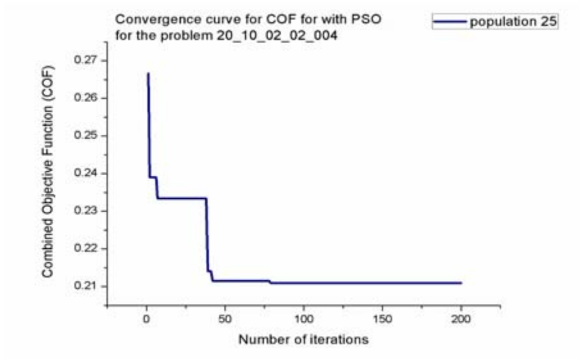

Figure-1. Convergence curve for COF with PSO for the problem 20_10_02_02_004.

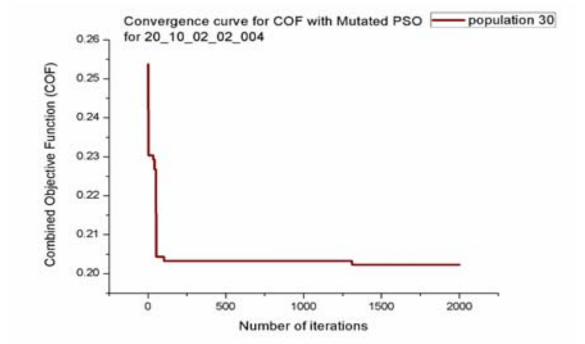

Figure-2. Convergence curve for COF with MPSO for the problem 20_10_02_02_004.

Table-4. COF value comparison.

m	n	Benchmark Problem	PSO	MPSO
2	20	20_02_02_02_001	0.30021	0.30021
2	20	20_02_02_02_002	0.3	0.3
3	20	20_03_02_02_001	0.24209	0.22716
3	20	20_03_02_02_002	0.22648	0.22644
3	20	20_03_02_10_003	0.21732	0.21693
3	20	20_03_02_10_004	0.21684	0.19235
3	20	20_03_04_02_001	0.31709	0.30006
3	20	20_03_04_02_002	0.27324	0.29664
3	20	20_03_04_02_003	0.3181	0.29973
3	20	20_03_04_02_004	0.29695	0.29542
3	20	20_03_04_02_005	0.27766	0.27208

3	20	20_03_04_04_001	0.29293	0.27086
3	20	20_03_04_04_002	0.26791	0.24664
3	20	20_03_04_04_003	0.29622	0.29193
3	20	20_03_04_04_004	0.30003	0.29184
3	20	20_03_04_04_005	0.26751	0.24302
3	20	20_03_04_06_001	0.29167	0.24302
3	20	20_03_04_06_002	0.24291	0.23337
3	20	20_03_04_06_003	0.29193	0.26888
3	20	20_03_04_06_004	0.29235	0.26735
3	20	20_03_04_06_005	0.24251	0.24201
3	20	20_03_04_08_001	0.24335	0.24205
4	20	20_04_02_02_001	0.2064	0.20514
4	20	20_04_02_02_002	0.20666	0.19659
4	20	20_04_02_02_003	0.20605	0.20566
5	20	20_05_02_02_001	0.18919	0.18835
6	20	20_06_02_02_001	0.17408	0.17114
7	20	20_07_02_02_001	0.18564	0.1827
8	20	20_08_02_02_001	0.20099	0.19551
9	20	20_09_02_02_001	0.19763	0.17179
10	20	20_10_02_02_001	0.19301	0.17893
10	20	20_10_02_02_002	0.21663	0.20389
10	20	20_10_02_02_003	0.19208	0.19052
10	20	20_10_02_02_004	0.210977	0.20228
10	20	20_10_02_02_005	0.23325	0.22513
10	20	20_10_02_04_001	0.21549	0.19763
10	20	20_10_02_04_002	0.19115	0.19115
10	20	20_10_02_04_003	0.19754	0.16552
10	20	20_10_02_04_004	0.20432	0.17625
10	20	20_10_02_04_005	0.20013	0.19962
10	20	20_10_02_06_001	0.17095	0.17095
10	20	20_10_02_06_002	0.17201	0.16663
10	20	20_10_02_06_003	0.18545	0.16201
10	20	20_10_02_06_004	0.1946	0.1946
10	20	20_10_02_06_005	0.20736	0.20736
10	20	20_10_02_08_001	0.19595	0.17095
10	20	20_10_02_08_002	0.15922	0.14222
10	20	20_10_02_08_003	0.20538	0.18428
10	20	20_10_02_08_004	0.23035	0.21551
10	20	20_10_02_08_005	0.21497	0.20482

4. CONCLUSIONS

The work presented in this paper is dealt with bi-objective Identical Parallel Machine Scheduling problem including the minimization of makespan and number of tardy jobs simultaneously. The problem belongs to NP hard and different meta-heuristics have been analyzed for IPMS problem. The Combined Objective Function (COF) proposed for scheduling the jobs in IPMS problem confirms to be an effective and comprehensive measure, as multiple decisions are frequently involved in this dynamic and competitive environment. To deal with conflicting criterion, it is also flexible as the optimal sequence can be obtained by varying the set of parameters considered. For this bi-objective IPMS problem, the meta-heuristics PSO and MPSO are proposed and analyzed the effectiveness through the proposed algorithm and it is found that the MPSO gives the better results when compared with PSO.

ACKNOWLEDGEMENTS

The Authors submit their thanks to Department of Mechanical Engineering, Kalasalingam University for their support to carry out the work.

REFERENCES

[1] Michael L. Pinedo: Scheduling: Theory, Algorithms and Systems. Springer, 2012.

[2] Ahmet B.Keha., Ketan Khowala., Fowler, J.W., “Mixed integer programming formulations for single machine scheduling problems”, Computers & Industrial Engineering 56, pp.357-367, 2009.

[3] Barnes., JW and Brennan., JJ., “An improved algorithm for scheduling jobs on identical machines”, AIIE Transactions., 9(1): pp.23–31., 1977.

[4] Dogramaci A., “Production scheduling of independent jobs on parallel identical machines”, International Journal of Production Research 16, pp.535–548, 1984.

[5] Elmaghraby SE and Park SH., “Scheduling jobs on a number of identical machines”, AIIE Transactions 6, pp.1–13, 1974.

[6] Azigzoglu M and Kirca O., “Tardiness minimization on parallel machines”, International Journal of Production Economics, 55(2): 163–168, 1998.

[7] Yalaoui F and Chu C., “Parallel machine scheduling to minimize total tardiness”, International Journal of Production Economics, 76(3): 265–279, 2002.

[8] Tanaka S and Araki M., “A branch-and-bound algorithm with Lagrangian relaxation to minimize total tardiness on identical parallel machines”, International Journal of Production Economics, 113(1): 446–458, 2008.

[9] Gupta JND and Ruiz Torres AJ., "A LISTFIT heuristic for minimizing makespan on identical parallel machines", Production Planning and Control 12(1): 28–36, 2001.

[10] Kennedy J and Eberhart RC. "Particle swarm optimization", In: Proceedings of IEEE international conference neural networks, Piscataway, NJ, 27 November-1 December 1995, pp.1942–1948. New York: IEEE.

[11] Liao CJ, Tseng CT and Luarn P., "A discrete version of particle swarm optimization for flowshop scheduling problems", Computers & Operations Research, 34(10): 3099–3111, 2007.

[12] Liu B, Wang L and Jin YH., "An effective hybrid PSO based algorithm for flow shop scheduling with limited buffers", Computers & Operations Research, 35(9): 2791–2806, 2008.

[13] Niu Q, Zhou T and Wang L., "A hybrid particle swarm optimization for parallel machine total tardiness scheduling", International Journal of Advanced Manufacturing Technology, 49: 723–739, 2010.

[14] Tasgetiren MF, Liang YC, Sevkli M., "A particle swarm optimization algorithm for makespan and total flowtime minimization in permutation flowshop sequencing problem", European Journal of Operational Research, 177(3): 1930–1947, 2007.

[15] Michalewicz Z. Genetic algorithms + data structures= evolution programs. 3rd ed. London: Springer-Verlag, 1996.

[16] Fisher ML., "A dual algorithm for the one-machine scheduling problem", Math Program 1976; 11: 229–251.

[17] Bathrinath S., Saravanasankar S., Mahapatra SS., Manas Ranjan Singh and Ponnambalam SG., "An improved meta-heuristic approach for solving identical parallel processor scheduling problem", Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture, February 17, 2015, doi: 0954405414564410.

[18] S.Bathrinath, S.Saravanasankar, SG. Ponnambalam and I.Jerin Leno, "Multi-objective Optimization in Identical Parallel Machine Scheduling Problem using NSGA-II", International Journal of Applied Engineering Research, ISSN 0973-4562, Volume 10, Number 6, pp.5050-5054, 2015.