Vol.65: e22210840, 2022
https://doi.org/10.1590/1678-4324-2022210840
ISSN 1678-4324 Online Edition

Article - Engineering, Technology and Techniques

Testing the Performance of Bat-Algorithm for Permutation Flow Shop Scheduling Problems with Makespan Minimization

Jeen Robert Bellabai [1*]
https://orcid.org/0000-0003-0109-577X

Brabin Nivas Murugadhas Leela[2]
https://orcid.org/0000-0003-1868-1118

Senthil Maharaj Ramesh Kennedy[3]
https://orcid.org/0000-0003-4932-0403

[1*,3]AAA College of Engineering and Technology, Department of Mechanical Engineering, Sivakasi, Tamilnadu, India; [2]Good Shepherd College of Engineering and Technology, Department of Mechanical Engineering, Maruthamparai, Tamil Nadu, India.

Editor-in-Chief: Alexandre Rasi Aoki
Associate Editor: Fabio Alessandro Guerra

Received: 29-Dec-2021; Accepted: 08-Mar-2022.

*Correspondence: jeenrobert@aaacet.ac.in; Tel.: +91-9488081212 (R.B.J.R.).

HIGHLIGHTS

- BAT algorithm is presented to minimize makespan as objective for the permutation flowshop scheduling problem by modifying the process parameter.
- In order to improve the initial quality & diversity, an NEH heuristics based constructive approach is given an initial solution.

Abstract: In this work, a BAT Algorithm is proposed to solve the permutation flow shop scheduling problem (PFSSP) with minimizing makespan criterion. In a PFSSP, there are n-jobs and m-machines with a proportional deterioration is considered in which all machines process the jobs in the same order, i.e., a permutation schedule. Every job comprises of a foreordained arrangement of assignment operations, each of which should be handled without intrusion for a given timeframe on a given machine. As of late, optimization algorithms such as ant colony optimization (ACO), simulated annealing (SA), artificial bee colony (ABC), genetic algorithm (GA), particle swarm optimization (PSO) and tabu search (TS) have assumed a significant role in solving PFSSPs. The popular NEH algorithm is considered as the parent algorithm to find the initial solution, and the makespan is minimized in two stages of simulation. The proposed algorithm is tested on 12 flow shop scheduling bench mark problems from OR Library. The proposed algorithm is validated with a well-chosen set of benchmark problems in the literature. Computational results indicate that the proposed bat algorithm is more efficient than the TLBO & HPSO algorithm.

Keywords: Scheduling; Flow shop; BAT algorithm; Makespan; Benchmark Problem.

INTRODUCTION

The permutation flow shop scheduling problem is a type of combinational optimization problem. In flow shop environment, there are 'n' jobs are to be processed through 'm' machines in the same order of sequence. In the PFSP, minimizing makespan is the most common objective. The complexity of job shop & flow shop scheduling problem with makespan objective has been solved by Garey and coauthors [1]. The first research on PFSP was proposed by Johnson [2] has become most challenging task for the researchers and practitioners. To solve this kind of complex PFSP both heuristic and meta-heuristic algorithm are to be used.

In early decades, many heuristic approaches has been developed for solving flow shop problems, e.g,, NEH heuristics [3],SPIRIT heuristics [4], Palmer heuristics [5], Hybrid Algorithm [6], Genetic Algorithm [7], Gupta heuristics [8]. Above all heuristics approach NEH heuristics is best constructive method for solving PFSP. Recently, many meta-heuristic algorithm are developed for solving Permutation flow shop scheduling problems; Rajkumar and coauthors [9] described an IGA algorithm to solve the PFSP. Wang and coauthors [10] developed a Diversity enhanced particle swarm optimization for solving flowshop benchmark problems. Multi-objective optimization using hybrid algorithm was proposed by Wang and coauthors [11] for the blocking permutation flow shop scheduling environment. Mostafa Akhshati and coauthors [12] presented hybrid algorithm for multi-objective optimization for solving flow shop scheduling problems. In this way, Adil baykasoglu and coauthors [13] developed teaching learning based optimization (TLBO) algorithm for solving large sized flow shop benchmark problems.

Then again, SFA is a well-known algorithm that recreates heredity of a sheep flock in a prairie. It is additionally an effective calculation to adapt to huge scale issues. In this context, Nara and coauthors [14] developed a new evolutionary algorithm based on sheep flocks heredity algorithm for solving any kind of scheduling problems, especially in various scheduling problems such as hybrid job shop scheduling and flow shop scheduling problems was solved by using improved sheep flock heredity algorithm proposed by Anandaraman [15]. Although a large number of optimization methods have been proposed for solving PFSSP, none of them is able to solve all instances of this problem. Each of them has a PFSSP in which it is ineffective or inapplicable. Therefore to solve these hard problems, investigations were carried out and the solutions are discussed in this paper. To the best of our insight, this is the primary examination which tries to build up these arrangement philosophies for minimizing makespan for flow shop scheduling problem.

In this paper, BAT algorithm is presented to minimize makespan as objective for the permutation flowshop scheduling problem by modifying the process parameter. In order to improve the initial quality & diversity, an NEH heuristics based constructive approach is given an initial solution.

This paper is organized as follows: the definition of permutation flow shop scheduling problem is described in section 2. The BAT is explained in section 3. Experimental results are presented in section 4. Conclusion and future are given in section 5.

PROBLEM DEFINITION

The permutation flow shop scheduling problem consisting of 'n' jobs, i.e., $\sigma = \{\sigma_{1,}\sigma_2, \dots\dots, \sigma_n\}$are to be processed through 'm' machines in the same order of job sequence. The main objective of PFSP is to determine the best job sequence i.e., $\sigma^* = \{\sigma_1^*, \sigma_2^*, \dots., \sigma_n^*\}$that minimizes the total completion time. Each job has m operations. In flow shop, every machine can able to process only one job at any interval of time. At any instance, all the jobs have to follow the same job sequence order.

$$C(\sigma_1, 1) = t(\sigma_1, 1) \tag{1}$$

$$C(\sigma_j, 1) = C(\sigma_j - 1, 1) + t(\sigma_j, 1) \text{ Where j=2,3,....,n} \tag{2}$$

$$C(\sigma_1, k) = C(\sigma_1, k-1) + t(\sigma_1, k) \text{ Where k=2,3,....,m} \tag{3}$$

$$C(\sigma_j, k) = \max\left(C(\sigma_j - 1, k), C(\sigma_j, k-1)\right) + t(\sigma_j, k) \tag{4}$$

Objective function is to find $C(\sigma_n, m) = makespan\ (completion\ time\ of\ \sigma_n\ on\ the\ machine\ m)$.
The following notations are considered in this PFSP:

$t(\sigma_j, k)$= processing time for job j on machine k (j=1,2,3,….,n), (k=1,2,3,….,m).
n = total number of jobs to be processed.
m= total number of machines in the process.
$\sigma = \{\sigma_{1,}\sigma_2, \dots\dots, \sigma_n\}$= permutation job set.

Assumptions Made

The following assumptions are made for the flow shop scheduling problem in this thesis:

- Each job ‘i’ can be processed at most on one machine ‘j’ at the same time.
- Each machine m can process only one job i at a time.
- No preemption is allowed, i.e. the processing of a job ‘i’ on a machine ‘j’ should not be interrupted.
- All the jobs are independent and they are available for processing at time 0.
- The set-up time of the jobs on machines is negligible and therefore, it can be ignored.
- The machines are continuously available.

The Proposed BAT Algorithm

BAT algorithm is a bio-inspired meta heuristic method based on the echolocation system of bats. Naturally bats emit ultrasonic pulses to the surrounding environment for hunting and navigation purposes. After these pulses emission, bats listen to the echoes which help them to locate themselves and also to locate and identify obstacles and preys. Furthermore, each bat of the swarm is able to identify the most “nutritious” areas performing an individual search, or moving towards a “nutritious” location previously found by the swarm. The main idea of the BAT algorithm is to imitate this echolocation system of the bats and some idealized rules have to be taken into account in order to make proper adaptations as proposed by Yang:

• All bats use echolocation to detect the distance, and they have one “magic ability” that allow them to differentiate a prey and an obstacle.

• All bats fly randomly with a velocity vi; at position xi; with a fixed frequency f_{min}, varying wavelength λ and loudness Ai to search a prey. In this idealized rule, we assume that every bat can adjust in an automatic way the frequency (or wavelength) of the emitted pulses, and the rate of these pulses emission r∈ [0,1] . This automatic adjustment depends on the proximity of the targeted prey.

• Generally the loudness of bats emissions can vary in many ways. Nonetheless, we assume that this loudness can vary from a large positive A0 to a minimum constant value Amin.

In Algorithm 1 the pseudocode of the basic BA is shown. On viewing this algorithm we can see that lines 1-6 correspond to the initialization process. At first the objective function has to be defined and the initial population has to be initialized. We assume that every bat of the population represents one possible solution to the addressed problem. Then, all the parameters related to each bat are initialized and defined. These parameters are the velocity vi, frequency fi, pulse rate ri and loudness Ai.

After these initialization steps, the algorithm starts its main phase. For each generation, every bat of the swarm moves by updating its velocity and position. For this movement, the following equations are used:

$$fi = fmin + (fmin - fmax)\beta$$

$$vi^{t} = vi^{t-1} + [xi^{t-1} - x*]fi$$

$$xi^{t} = xi^{t}-1 + vi^{t}$$

where the parameter β is a randomly generated number in the [0,1] interval. Additionally, x∗ denotes the current best solution in the swarm, and vt i and xt i represent the velocity and position of a bat i at time step t. Finally, the results of Equation (1) are used to control the range and pace of bats movement. In addition, for the local search part, whether a solution is selected among the best ones, a new solution for each bat is generated using a random walk

$$x_{new} = x_{old} + At$$

where is a randomly generated number within the interval [-1,1], and At is the average loudness of the swarm at time step t. Finally, the loudness Ai and the rate ri of each bat have to be updated if the conditions shown in the line 14 of Algorithm 1 are met. This update is conducted as follows:

$$At+1\ i = \alpha At+1\ i \tag{5}$$

$$rt+1\ i = r0\ i\ [1-exp(-\gamma t)] \tag{6}$$

where α and γ are constants. Thereby, for any 0<α<1 and γ>0 we have
at i → 0,rt i → r0 i , as t →∞ (7) In many studies of the literature, α = γ is used in order to simplify the implementation of the algorithm. Specifically, we use α = γ = 0.98 in this work. We have chosen this value empirically using a [0.90, 0.99] range.

Algorithm 1: Pseudocode of the basic BA

```
Define the objective function f(x);
Initialize the bat population X = x_1, x_2, ..., x_n;
for each bat x_i in the population do
    Initialize the pulse rate r_i, velocity v_i and loudness A_i;
    Define the pulse frequency f_i at x_i;
end
repeat
    for each bat x_i in the population do
        Generate new solutions through Equations 1, 2 and 3;
        if rand>r_i then
            Select one solution among the best ones;
            Generate a local solution around the best one;
        end
        if rand<A_i and f(x_i)<f(x_*) then
            Accept the new solution;
            Increase r_i and reduce A_i;
        end
    end
until termination criterion not reached;
Rank the bats and return the current best bat of the population;
```

Figure 1. Flow diagram of BAT

EXPERIMENTAL RESULTS AND DISCUSSION

A broad exploratory assessment and examination with a portion of the late techniques is given in light of an understood flow shop benchmark set of Taillard [16], comprising of 12 group of problems with the size going from "20 jobs 5 machines" to "500 jobs 20 machines". The coding for the optimization of scheduling has been created utilizing MATLAB 7.6.0 programming and tried on a 2.00GHz Core i3 PC. The populace size and the generation number are set to 150 and 1000, separately. Analyses are rehashed 10 times for each problem group. Computational results are given in Table 1. The sections in this table demonstrate the extent of every group (jobs machines), benchmark set of Taillard [16], the well-known solutions (wks) (known as optimal or the most minimal upper bound for Taillard's occasions), the best sequence or least makespan, standard deviation and Average Relative Percentage Deviation (ARPD) for the acquired results. Figure 2, showing the comparison of proposed BAT algorithm with TLBO [13] algorithm for the 12 Taillard instances (Ta 001, Ta 011, Ta 021, Ta 031, Ta 041, Ta 051, Ta 061, Ta 071, Ta 081, Ta 091, Ta 101 and Ta 111). ARPD from the well-known solution is processed as takes after:

$$(ARPD) = \sum_{i=1}^{R} \frac{(\,Best\ Solution_i - well\ known\ solution)}{well\ known\ solution} * 10^2 \,/\, R \qquad (7)$$

Where Best Solution is the makespan obtained by BAT algorithm in each run whereas Well Known Solution is the optimal or the lowest known upper bound for Taillard’s instances. Later, the execution of the proposed BAT is checked with TLBO algorithm. The test results demonstrates that the proposed BAT performs superior to anything TLBO in many occurrences. We additionally contrasted the BAT with the novel particle swarm optimization algorithm (NPSO) proposed by Lian and coauthors [18] and the hybrid particle swarm optimization (HPSO) was proposed by Kuo and coauthors [19] calculation since these calculations have as of now been appeared to be better than the well-known solutions in the literature. As it can be seen from Table 2, the execution of the BAT algorithm is equivalent with the best-known solutions from the literature. Despite the fact that the BAT did not create the best results when contrasted with the well-known solutions and algorithms, its outcomes are near those of well-known solutions and/or similar. It appears that in its essential structure (as we deliberately actualized it in its fundamental structure keeping in mind the end goal to better assess its genuine execution) the BAT can possibly give great solutions for the FSSP. For the 12 benchmark problem verified by TLBO algorithm instances, the proposed BAT obtained the most superior solution in 10 instances. From our experiment, it is observed that the BAT is competitive.

An extra near study is done for testing the productivity of the BAT algorithm with two variants of the iterated greedy algorithm (IG1 and IG2) (which was proposed by Ribas and coauthors [17]) and the hybrid discrete differential evolution algorithm (HDDE) (which was proposed by Wang and coauthors [11] taking into account ARPD estimations) and Teaching Learning Based Optimization (TLBO) calculation (which was proposed by Adil Baykasoglu and coauthors [13]) is likewise performed. Table 3 demonstrates that, all things considered, the BAT algorithm perform superior to the TLBO, HDDE algorithm furthermore superior to the IG1 and the IG2 algorithms. Figure 3 shows the Average Relative Percentage Deviation of all algorithm with 12 Taillard's Benchmark problems set.

Table 1. Computational results for FSSP test problems

TLBO for FSSP	Prob	WKS	BAT				TLBO [13]			
			Best	**Max**	**Avg**	**APRD**	**Best**	**Max**	**Avg**	**APRD**
20x5	Ta 001	1278	**1278**	1297	1284.9	0.5399	1278	1297	1287.2	0.7199
20x10	Ta 011	1582	1609	1683	1623.3	2.6106	1586	1618	1606	1.5171
20x20	Ta 021	2297	**2323**	2401	2355.4	2.5424	2325	2370	2344.7	2.0766
50x5	Ta 031	2724	**2724**	2730	2725.6	0.0587	2724	2741	2729.4	0.1982
50x10	Ta 041	2991	**3119**	3145	3110.6	3.8449	3120	3169	3141.4	5.0284
50x20	Ta 051	3771	4001	4101	4021.9	6.6534	3986	4095	4029.7	6.8602
100x5	Ta 061	5493	**5493**	5519	5496.4	0.0619	5493	5527	5499.4	0.1165
100x10	Ta 071	5770	**5808**	5846	5819.6	0.8596	5887	5997	5928.7	2.7504
100x20	Ta 081	6286	**6485**	6607	6527.2	3.8371	6549	6726	6617.8	5.2784
200x10	Ta 091	10868	**10942**	10942	10942	0.6809	10979	11079	11033	1.5182
200x20	Ta 101	11294	**11600**	11639	11622.5	2.9086	11855	12024	11940	5.7199
500x20	Ta 111	26189	**26612**	26652	26622.6	1.6557	27377	27565	27492	4.9754

Best results obtained by BAT is given as Bold.

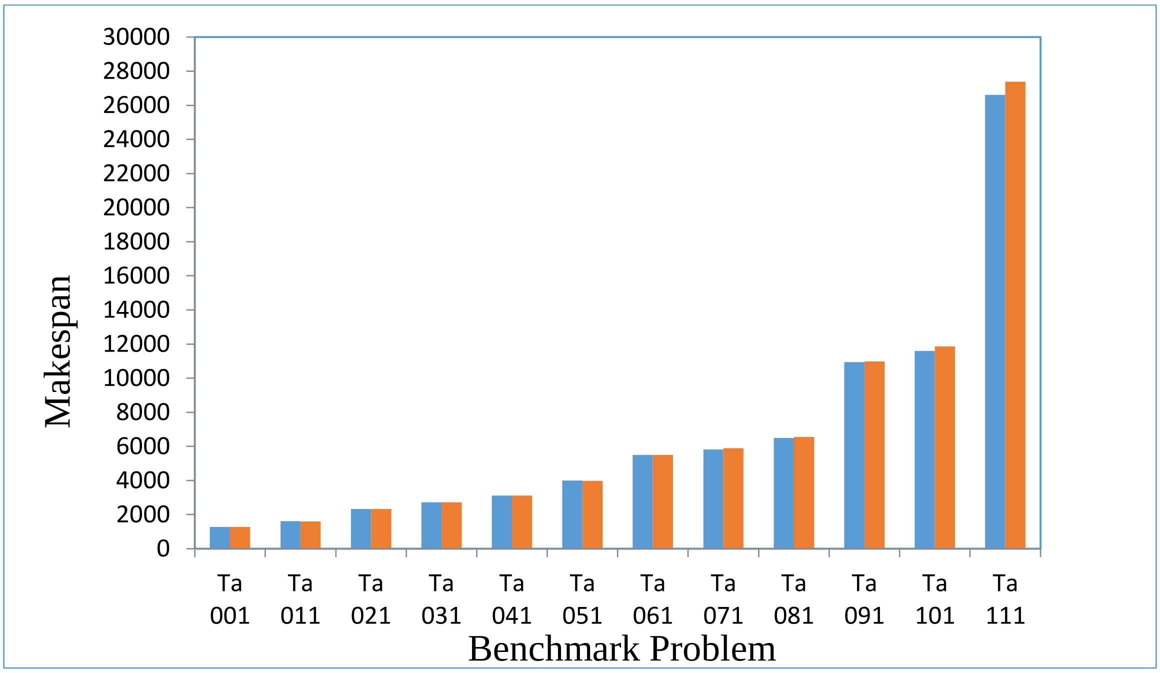

Figure 2. Chart showing the comparison of BAT algorithm with TLBO [13]

Table 2. Comparison of BAT with some novel algorithms for FSSP test problems.

Prob	PS (J*M)	WKS	TLBO[13]			HPSO[18]			NPSO [19]			BAT		
			Best	Max	Avg	Best	Max	Avg	Best	Max	Avg	Best	Max	Avg
Ta 001	20x5	1278	1278	1297	1287.2	1278	1278	1278	1278	1297	1279.9	**1278**	1297	1284.9
Ta 011	20x10	1582	1586	1618	1606	1582	1596	1587.3	1582	1639	1605.8	1609	1683	1623.3
Ta 021	20x20	2297	2325	2370	2344.7	2297	2315	2307	2297	2376	2334.9	2328	2401	2355.4
Ta 031	50x5	2724	2724	2741	2729.4	2724	2724	2724	2724	2729	2725	**2724**	2730	2725.6
Ta 041	50x10	2991	3120	3169	3141.4	3034	3063	3053.6	3034	3129	3086.9	3078	3145	3110.6
Ta 051	50x20	3771	3986	4095	4029.7	3923	3963	3944.6	3938	3989	3964.3	4001	4101	4021.9
Ta 061	100x5	5493	5493	5527	5499.4	5493	5493	5493	5493	5495	5493.2	**5493**	5519	5496.4
Ta 071	100x10	5770	5887	5997	5928.7	None	None	None	None	None	None	**5808**	5846	5819.6
Ta 081	100x20	6286	6549	6726	6617.8	None	None	None	None	None	None	**6485**	6607	6527.2
Ta 091	200x10	10868	10979	11079	11033	None	None	None	None	None	None	**10942**	10942	10942
Ta 101	200x20	11294	11855	12024	11940	None	None	None	None	None	None	**11600**	11639	11622.5
Ta 111	500x20	26189	27377	27565	27492	None	None	None	None	None	None	**26612**	26652	26622.6

Best results obtained by BAT is given as Bold.

Table 3. ARPD on Taillard instances for each algorithm

Prob	PS(J*M)	IG1[17]	IG2[17]	HDDE [11]	TLBO	BAT
Ta 001	20x5	0.39	0.46	1.49	0.72	0.54
Ta011	20x10	0.48	0.62	1.53	1.52	2.61
Ta 021	20x20	0.31	0.32	1.23	2.08	2.54
Ta 031	50x5	2.71	2.99	5.69	0.20	**0.06**
Ta 041	50x10	3.24	3.23	5.63	5.03	4.00
Ta 051	50x20	2.88	2.54	5.04	6.86	6.65
Ta 061	100x5	3.82	3.56	7.22	0.12	**0.06**
Ta 071	100x10	3.34	3.48	6.67	2.75	**0.86**
Ta 081	100x20	3.03	2.82	4.41	5.28	3.84
Ta 091	200x10	3.85	3.63	6.91	1.52	**0.68**
Ta 101	200x20	2.31	2.2	4.34	5.72	2.91
Ta 111	500x20	1.32	1.33	3.93	4.98	1.66
AREP		2.31	2.27	4.51	3.07	**2.20**

Best results obtained by BAT is given as Bold. AREP: Average relative error percentage

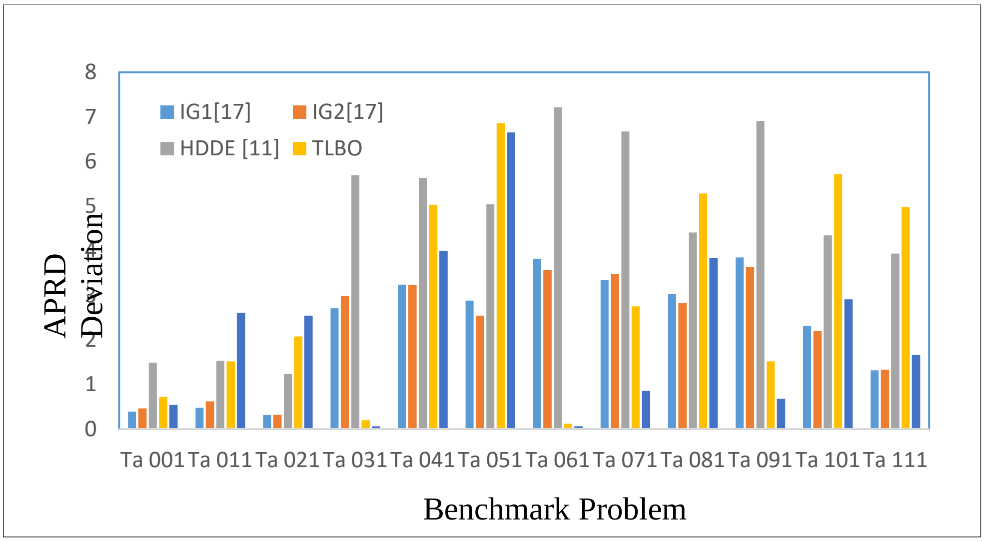

Figure 3. Average Relative Percentage Deviation of all algorithm with Benchmark problems

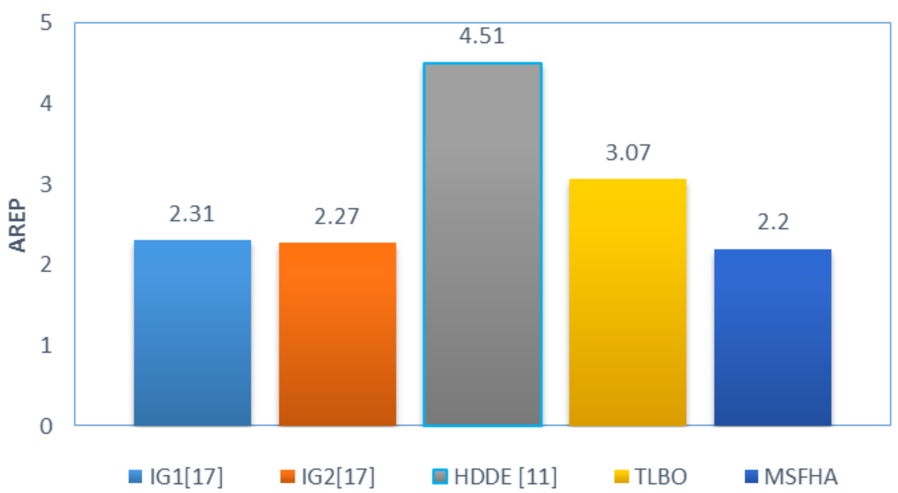

Figure 4. Comparison of AREP of BAT for makespan

Moreover, the average REP(AREP) of the proposed BAT algorithm is also (2.2) lesser than that of all other approaches such as iterated greedy algorithm (IG1 and IG2), hybrid discrete differential evolution algorithm (HDDE), and Teaching Learning Based Optimization (TLBO), as shown in Figure 4. Computational results describe that the proposed BAT gives better quality solutions for the measure of makespan minimization in the extensive estimated flow shop scheduling benchmark problems.

CONCLUSION AND FUTURE WORK

In this paper, BAT algorithm is depicted for getting the optimal ideal solution for flow shop scheduling problem with the consideration of minimizing makespan as objective. The execution of the proposed algorithm has been affirmed with the results available in the literature. Moreover the genuine duty of this work has been the use of a meta-heuristic algorithm considered for handling the flow shop scheduling substantial measured Taillard benchmark problem (20 jobs, 5, 10, 20 machines and 50 jobs, 5, 10, 20 machines and 100 jobs, 5, 10 and 20 machines and 200 jobs, 10, 20 machines and 500 jobs, 20 machines). The exploratory results demonstrate that the proposed HGASA algorithm, by actualizing scramble and swap mutation additionally robust-replace heuristic is productive in finding worldwide best solution for the permutation flow shop scheduling problems with a goal of makespan minimization. Computational results demonstrate that the proposed MSFHA gives noteworthy better-quality solutions for the measure of makespan minimization in the extensive estimated flow shop scheduling benchmark problems. As a future work, the proposed algorithm can be connected to take care of job shop scheduling problems and flexible flow shop scheduling problems. Besides, BAT algorithm can be hybrid with other optimization algorithm for better merging to achieve global best solution for the multiple objectives problem.

Conflicts of Interest: The authors declare no conflict of interest.

REFERENCES

1. Garey MR, Johnson DS, Ravi Sethi. The Complexity of Flowshop and Jobshop Scheduling. Math. Oper. Res. 1976; 1(2):117-29.
2. Johnson SM. Optimal two-and three-stage production schedules with setup times included. Nav. Res. Logist. Q. 1954;1(1):61-8.
3. Nawaz M, Enscore J, Ham I. A Heuristic algorithm for the M machine, n-job flow shop sequencing problem. OMEGA. 1983;11(1):91-5.
4. Widmer M, Hertz A. A new heuristic for the flow-shop sequencing problem. Eur. J. Oper. Res. 1989; 41: 186-93.
5. Palmer DS. Sequencing jobs through a multi-stage process in the minimum total time: a quick method of obtaining a near optimum. Oper. Res. 1965; 16(1): 101-7.
6. Jeen Robert RB, Rajkumar R. A Hybrid Algorithm for Minimizing Makespan in the Permutation Flow Shop Scheduling Environment. Asian J. Res. Soc. Sci. Humanit. 2016;6(9):1239-55.
7. Jeen Robert RB, Rajkumar R. An Effective Genetic Algorithm for Flow Shop Scheduling Problems to Minimize Makespan. Mechanika. 2017; 23(4): 594-03.
8. Gupta JN. A functional heuristic algorithm for the flow-shop scheduling problem. Oper. Res.1971; 22(1): 39-47.
9. Rajkumar R, Shahabudeen P. Bi-criteria Improved Genetic Algorithm for Scheduling in Flow shops to Minimize the Makespan and Total Flowtime of Jobs. Int. J. Computer Integr. Manuf., 2009; 22(10): 987-98.
10. Wang L, Zheng DZ. An Effective Hybrid Heuristic for Flow Shop Scheduling. Int. J. Adv. Manuf. Technol., 2003; 21: 38-44.
11. Wang H, Sun H, Li C, Rahnamayan S, Pan JS. Diversity enhanced particle swarm optimization with neighborhood search. Inf. Sci., 2013;223:119-35.
12. Mostafa Akhshabi, Javad Haddadniab, Mohammad Akhshabi. Solving flow shop scheduling problem using a parallel genetic algorithm. Proc. Technol. 2012;1:351-55.
13. Adil Baykasoğlu, Alper Hamzadayi, Simge Yelkenci Köse. Testing the performance of teaching–learning based optimization (TLBO) algorithm on combinatorial problems: Flow shop and job shop scheduling cases. Inf. Sci., 2014; 276: 204-18.
14. Nara K, Takeyama T, Hyungchul Kim. A new evolutionary algorithm based on sheep flocks heredity model and its application to scheduling problem. IEEE Trans. Syst. Man Cybern.: Syst. 1999;6:503-8.
15. Anandaraman. An improved sheep flock heredity algorithm for job shop scheduling and flow shop scheduling problems. Int. J. Ind. Eng. Comput. 2011; 2: 749–64.
16. Taillard E. Benchmarks for basic scheduling problems. Eur. J. Oper. Res. 1993; 64; 278-85.
17. Ribas I, Companys R, Tort-Martorell X. An iterated greedy algorithm for the flowshop scheduling problem with blocking. Ómega. 2011; 39: 293–301.

18. Lian Z, Gu X, Jiao B. A novel particle swarm optimization algorithm for permutation flow-shop scheduling to minimize makespan. Chaos Solitons Fractals. 2008; 35: 851–61.
19. Kuo I, Horng S, Kao T, Lin T, Lee C, Terano T, Pan Y. An efficient flow-shop scheduling algorithm based on a hybrid particle swarm optimization model. Expert Syst. Appl. 2009; 36: 7027–32.