Bat Algorithm for Job Shop Scheduling Problem

Hegazy Zaher,
Prof. of Statistics

Mahmoud El-Sherbieny,
Prof. of Operations Research

Naglaa Ragaa Saied,
Assoc. Prof. of Operations Research

Heba Sayed,
Candidate of Doctoral in Operations Research

Institute of Statistical Studies & Researches, Cairo University, Egypt.

***Abstract*— The job shop scheduling problem is one of the most important in manufacturing planning. It is one of the most difficult NP-hard and combinatorial problems. In the past, the exact methods are guaranteed to find the optimal solution for small problems but they are useless for large problems. Recently, the approximation methods are used as an alternative to the exact methods for solving NP-hard problems. In this paper, a proposed bat algorithm is introduced to solve the job shop scheduling problem. Based on ten benchmark problems, results demonstrate that the proposed algorithm gives better results than the particle swarm algorithm in both convergence speed and accuracy.**

***Keywords*— Bat Algorithm, Job Shop Scheduling, Makespan, Giffler and Thompson Algorithm.**

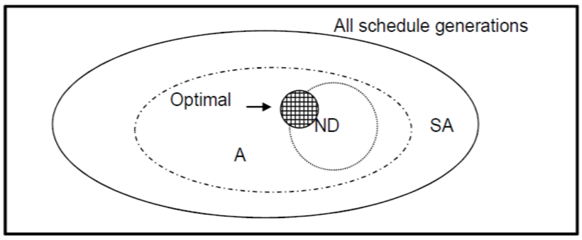
I- INTRODUCTION

Scheduling problems are one of the most important problems in the field of combinatorial optimization and their applications in various engineering and manufacturing industries [1].

Scheduling is defined as the process of assigning a set of tasks to resources over a period of time or it may be defined as the allocation of resources over time to perform a collection of tasks.

The Job Shop Scheduling Problem (JSSP) is one of the most popular and generalized production systems, which are hard to solve thanks to their non-polynomial hard nature.

There are three major kinds of the feasible schedule; they are semi-active, active and non-delay. The semi-active schedule is the schedule, where it is not possible to schedule the operation earlier without changing the sequence in which they are entering the machine. The active schedule is than the schedule, where is not possible to create the schedule by changing the order of the operation by starting the operation earlier without delaying other one. This schedule generation is the most used in the optimization because the optimal schedule is always the active one same as the Semi-Active and in the same times it is the subset of the semi-active schedules. So it gives us much smaller searching the neighborhood to search than the Semi-Active ones. The last mentioned schedule generation is non-Delay, which is the subset of the active schedules (see Fig. 1). In this schedule no machine is idle (without assigning job), when the operation is available [2]. The Job Shop Scheduling Problem is one of the most important industrial activities, especially in manufacturing planning.



SA – Semi-Active schedules; A - Active schedules; ND – Non-Delay schedules

Fig.1 Schedule generation map [1].

It is one of the most difficult NP-hard and combinatorial problems. In the past, numeration studies showed that exact methods are guaranteed to find the optimal solution for small problems but they are useless for large problems. Recently, the approximation methods are used as an alternative to the exact methods for solving NP-hard problems. The approximation methods are classified as heuristics and meta-heuristics.

In recent years, using meta-heuristic methods to solve JSSP has been growing rapidly, such as Genetic algorithms (GA) are proposed for solving JSSP as in [3-7], Ant Colony Optimization (ACO) is presented to minimize makespan for JSSP as in [8-12], Particle Swarm Optimization (PSO) is proposed as in [13-16], Tabu Search (TS) is used for solvine JSSP as in [17-19], Simulated Annealing (SA) is presented as in [20-23], hybrid PSO is presented as in [24], hyprid GA is as in [25] and hybrid swarm intelligence algorithmis as [26].

Bat Algorithm (BA) was proposed by Yang (2010). It is a new meta-heuristic optimization algorithm observing and searching for the prey of the bats. The advantage of BA is that it can provide very quick convergence at a very initial stage by switching from exploration to exploitation[27]. It is potentially more powerful than PSO and GA. The primary reason in using BA is a good combination of major advantages of these algorithms in some way. Moreover, PSO is the special case of the BA under appropriate simplifications.

In this paper, BA is applied to solve the JSSP. The optimal JSSP solution should be an active schedule, thus, developed Giffler and Thompson's heuristic is applied to decode a bat position into a schedule.

This paper is structured as follows. In Section II, presents "Methodology". In Section *A* introduces "Job Shop Scheduling Problem Formulation". In Section *B*, introduces "The Bat Algorithm". In Section *C*, introduces "Priority-based representation". In section *D*, introduces "Giffler and Thompson Algorithm". In section III, presents "Proposed Bat Algorithm for Job Shop Scheduling Problem". In Section IV, the proposed bat algorithm is tested on Fisher and Thompson (1963) and Lawrence (1984) test problems. Finally, conclusion is given in Section V.

II- METHODOLOGY

A. Job Shop Scheduling Problem Formulation

JSSP is defined as following: - There are a job set $J = \{J_1, J_2, ..., J_n\}$ and a machine set $M = \{M_1, M_2, . . ., M_m\}$. Each job, J_n, must be preformed through *m* machines to complete its work. Each job comprises of a set of operations, and the operation order for the machines is predefined. Each operation uses one of machines to complete its work for a fixed time interval. Once an operation is processed on a given machine, it cannot be interrupted before it finishes the procedure. The sequence of the operations of a job should be predetermined and may be different for any job. Each job has a sequence of operations. Each machine can process only one operation during the time interval. The objective of the JSSP is to find an appropriate schedule. A good schedule is a suitable operation planning for all jobs that can minimize the makespan or one that minimizes the idle time of machines [28].

The makespan is denoted as C_{max}. It is the maximum total completion time of the latest operation in the schedule of $n \times m$ operations.

The general job shop scheduling mathematical model as presented in [29].The detail of machine availability constraint and variable are presented as follows:

Let $t_{i,j}$ Be start time of job *j* that is performed on the machine *I*,

Let $f_{i,j}$ be finish time of job *j* that is performed on machine *i*,

Let $P_{i,j}$ be processing time of job *j* that is performed on machine *i*,

Let C_{max} be makespan (finish time of latest job).

The objective of the problem is to minimize makespan. The mathematical model of JSSP without machine availability constraint is shown below.

$$\text{Min} \quad C_{max} \tag{1}$$

St.

$$t_{h,j} - t_{i,j} \geq p_{i,j} \tag{2}$$

$$C_{max} - t_{i,j} \geq p_{i,j} \tag{3}$$

$$t_{i,j} - t_{i,k} \geq p_{i,k} \text{ or } \quad t_{i,k} - t_{i,j} \geq p_{i,j} \tag{4}$$

$$t_{i,j} \geq 0 \tag{5}$$

To make sure that the next step on machine h of job *j* starts after finish time of the step on machine *i* of job *j*, equation 2 is employed. Next, equation 3 ensures that C_{max} must be more than finish time of the last job. Equation 4 is used for sequencing jobs on the machines. This equation means that only one job can be processed only one machine at a time. By using equation 5, the start time of processes is non negative.

B. Bat Algorithm

BA is an evolutionary algorithm introduced by Yang. Three major characteristics of the microbat are employed to construct the basic structure of BA.

The used approximate and the idealized rules in Yang's method are listed as follows [27]:

- Most of the species of the bat utilize the echolocation to detect their prey, but not all species of the bat do the same thing. However, the micro bat is a famous example of extensively using the echolocation. Hence, the first characteristic is the echolocation behavior.
- The second characteristic is the frequency that the micro bat sends a fixed frequency Q_{min} with a variable wavelength λ and the loudness A_0 to search for prey.
- There are many ways to adjust the loudness. For simplicity, the loudness is assumed to be varied from a positive large A_0 to a minimum constant value, which is denoted by Amin. In Yang's method, the movement of the virtual bat is simulated by Eq. (1) – Eq. (3):

$$Q_i = Q_{min} + (Q_{max} - Q_{min}) * \beta \tag{1}$$

$$v_i^t = v_i^{t-1} + (x_i^t - x_{best}) * Q_i \tag{2}$$

$$x_i^t = x_i^{t-1} + v_i^t \tag{3}$$

Where (Q_i) is the frequency used by the bat seeking for its prey, the suffixes, min and max, represent the minimum and maximum value, respectively. x_i denotes the location of the i[th] bat in the solution space, v_i represents the velocity of the bat, t indicates the current iteration, β is a random vector, which is drawn from a uniform distribution, and $\beta \in [0, 1]$ and x_{best} indicates the global near best solution found so far over the whole population.

In addition, the rate of the pulse emission from the bat is also taken to be one of the roles in the process. The pulse emission rate is denoted by the symbol r_i, and $r_i \in [0, 1]$ where the suffix *i* indicates the i[th] bat. In every iteration, a random number is generated and is compared with r_i. If the random number is greater than r_i, a local search strategy, namely, random walk, is detonated. A new solution for the bat is generated by Eq. (4):

$$x_{new} = x_{old} + \varepsilon \overline{A^t} \tag{4}$$

Where ε is a random number and $\varepsilon \in [-1, 1]$ and $\overline{A^t}$ represents the average loudness of all bats at the current time step. After updating the positions of the bats, the loudness Ai and the pulse emission rate ri are also updated only when the global near best solution is updated and the random generated number is smaller than r_i. The update of A_i and r_i are operated by Eq. (5) and Eq. (6):

$A_i^{t+1} = \alpha * A_i^t$ (5)

$r_i^{t+1} = r_i^0[1 - e^{-\gamma t}]$ (6)

Where α and γ are constants. In Yang's experiments, α = γ = 0.9 is used for the simplicity. r_i^0 and A_i are factors which consist of random values.

Algorithm 1. Pseudo code of the BA [27]

1. Objective function: $f(x), x = (x_{1,} \dots., x_d)$
2. Initialize bat population x_i and velocity $v_i, i = 1,2, \dots, n$
3. Define frequency Q_i at x_i
4. Initialize pulse emission rate r_i and loudness A_i
5. **While** (t<maximum number of iterations)
6. Generate new solutions by adjusting frequency, and updating velocities and location/solutions.
7. **If** (rand > r_i)
8. Select a solution among the best solutions
9. Generate a local solution around the selected best solution
10. **End If**
11. **If** (rand< *Ai* and f(*x*i)< *f*(*x**))
12. Accept new solutions
13. Increase r_i reduce A_i
14. **End If**
15. Ranks the bats and find current best *x**
16. **End While**
17. Display results.

C. Priority-based Representation

When the BA is applied (i.e., the bats search solutions in a continuous solution space), each value of a bat position represents the associated operation priority. For an n-job m-machine problem, we can represent the bat k position by an *m* * *n* matrix, i.e.

$$X^k = \begin{bmatrix} x_{11}^k & x_{12}^k & \cdots & x_{1n}^k \\ x_{21}^k & x_{22}^k & \cdots & x_{2n}^k \\ \vdots & \vdots & & \vdots \\ x_{m1}^k & x_{m2}^k & \cdots & x_{mn}^k \end{bmatrix}$$

Where x_{ij}^k denotes the priority of operation o_{ij} and o_{ij} is the operation of job *j* that needs to be processed on machine *i*.

D. Giffler and Thompson Algorithm

A bat position can be mapped (or decoded) into an active schedule using Giffler and Thompson's heuristic. The Giffler and Thompson (G&T) algorithm is described as follows [30]:

Notation:

(*i*, *j*): the operation of job *j* that needs to be processed on machine *i*.

$T_{(j,\,i)}$: Job sequence matrix where *j* is job number and *i* is machine number.

$P_{(j,\,i)}$: Processing time matrix.

$X_{(i,\,j)}$: priorities matrix.

S: the partial schedule that contains scheduled operations.

U: the set of schedulable operations.

$s_{(i,j)}$: the earliest time at which operation (*i*, *j*) є *U* can be started.

$p_{(i,j)}$: the processing time of operation (*i, j*).

$f_{(i,j)}$: the earliest time at which operation (*i*, *j*) є *U* can be finished,

$f_{(i,j)} = s_{(i,j)} + p_{(i,j)}$.

G&T Algorithm 2:

Step 1: Initialize S = {}; *U* is initialized to contain all operations without predecessors.

Step 2: Determine $f^* = \min_{(i,j) є U}\{f_{(i,j)}\}$ and the machine *m** on which *f** could be realized.

Step 3:

(1) Identify the operation set (*i*′, *j*′) ∈ *U* such that (*i*′,*j*′) requires machine *m**, and $s_{(i',j')} < f^*$.

(2) Choose (*i*, *j*) from the operation set identified in (1) with the largest priority.

(3) Add (*i*, *j*) to *S*.

(4) Assign $s_{(i,j)}$ as the starting time of (*i*, *j*).

Step 4: If a complete schedule has been generated, stop. Else, delete (*i*, *j*) from *U* and include its immediate successor in U, then go to Step 2.

III- PROBOSED BAT ALGORITHM FOR JOB SHOPSCHEDULING PROBLEM

In previous researches, BA was used to solve continuous optimization problems. In JSSP, the solution space is discrete, thus, the priority based representation for bat algorithm is applied and then, the Giffler and Thompson algorithm and then, the swap operator is used to enhance bat solutions. In this section, describes how to combine between Proposed Bat Algorithm (PBA) and G&T algorithm to solve JSSP as following.

Proposed Bat Algorithm 3:-

1. **Initialization**: Generate bat population in continuous space.
2. **Representation of individuals**: Representation of bats in PBA for JSSP using priority based representation.

3. **Evaluation**: Evaluate the value of fitness function (makespan) using Giffler and Thompson algorithm.
4. **Update**: Update the velocity and bat positions using Eqs. (1), (2) and (3).
5. **Generate a local solution**: Generate a local solution around the selected best solution if the generated random number is greater than r_i using Eq. (4).
6. **Repeat step 3.**
7. **Update Process of Loudness and Pulse Emission Rate:**

   Loudness (A_i) and pulse emission rate (r_i) must be updated using Eqs. (5) and (6) only if the global best solution is updated and the randomly generated number is smaller than A_i.
8. **Swap operator**

   Swap operator is choosing two different positions from a job permutation randomly and swap them.
9. **Termination**: Repeat Step 2 to Step 9 until the predefined value of the fitness function is achieved or the maximum number of iterations has been reached. Record the best value of the fitness function and the best bat position among all the bats.

IV- COMPUTATIONAL RESULTS

The PBA were tested on (FT06, FT10, and FT20) [31], (LA01 to LA07) [32]. These problems are available on the OR-Library web site [33]. Algorithms are tested with 30 independent runs; the number of individual (bat) in population is fixed to 30.Maximum iterations for the priority-based IBA is set 500 for each run. Qmin is 0 and Qmax is 1 while α and γ are 0.9 for bat algorithm. The proposed algorithm is compared with the priority-based PSO (D.Y. Sha & Cheng-Yu Hsu, 2006). The computational results of FT and LA test problems are shown in Table 1. Table 1 includes (size of each problem, Best Known Solution (BKS) and (best solution, average and Relative Percentage Error (RPE) for each method)).

$$\text{RPE}=\frac{best-BKS}{BKS}*100$$

TABLE 1. COMPUTATIONAL RESULTS OF PSO and PBA

	problem	Size (n*m)	BKS	PSO-priority based			PBA-priority based		
				best	Average	RPE	best	Average	RPE
1	Ft06	6*6	55	55	58.9	0	55	**56.8**	0
2	Ft10	10*10	930	1007	1086	8.279569892	**1004**	1076.566667	7.956989247
3	Ft20	20*5	1165	1242	1296	6.60944206	**1203**	1283.7	3.2618026
4	La01	10*5	666	681	705	2.252252252	**666**	695.9	0
5	La02	10*5	655	694	729.7	5.954198473	**672**	696.9666667	2.595419847
6	La03	10*5	597	633	657.5	6.030150754	**621**	633.4666667	4.020100503
7	La04	10*5	590	611	648.1	3.559322034	**610**	633.3	3.3898305
8	La05	10*5	593	593	601.1	0	593	**599.8**	0
9	La06	15*5	926	926	940.2	0	926	**938.5**	0
10	La07	15*5	890	890	940.1	0	890	**934.9333**	0

The comparison is based on the results for the problems of Fisher and Thompson shown in Table 1 and Fig. 2. It can be observed that two algorithms generated the best known solution for the FT06 problem. For the remaining two problems (FT10 and FT20) from first type of bench mark problems, that two algorithms do not give the best known solution. But PBA gives result better than PSO in two problems. Next comparison is based on the Lawrence problems. PBA is able to find the best known solution (BKS) for four problems (La01, La05, La06 and La07) out of 7. PBA is able to find the results are better than PSO in three problems (La02, La03, and La04)

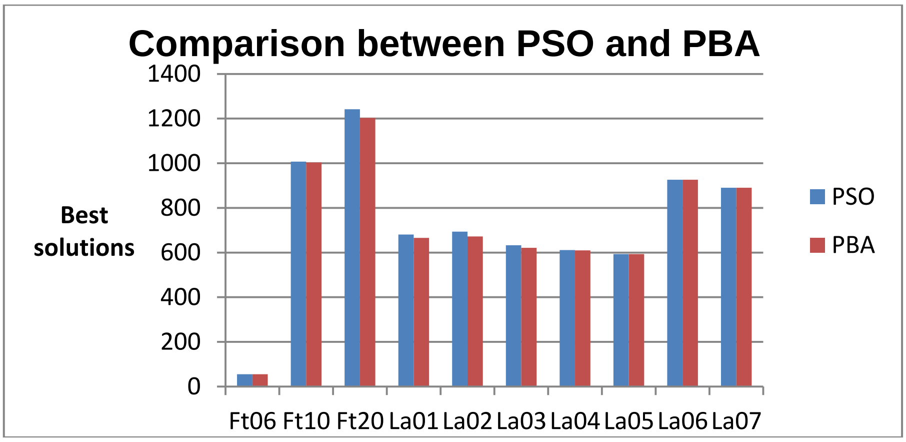


Fig. 2 Comparison between PSO and PBA using best solution

Comparison between PSO and IBA is shown in Fig. 3. Relative percent error (RPE) for both the IBA and PSO algorithms is zero for problems (Ft06, La05, La 06 and La07) since both the algorithms are able to find the best known solutions for the 4 problems. IBA gives better REP for problems (Ft10, Ft20, La01, La02, La03, and La04) as compared to those given by PSO method.

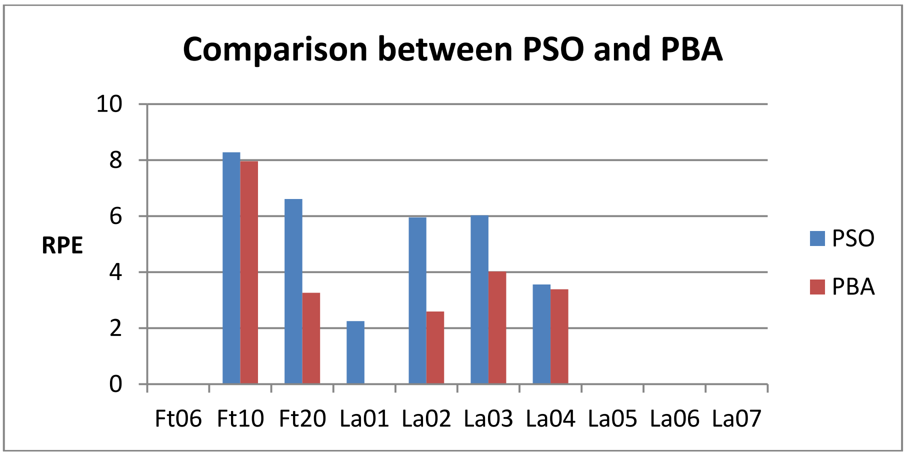


Fig. 3 Comparison between PSO and PBA using RPE

V- CONCLUSION

The improved bat algorithm is given for solving job shop scheduling problem. The performance of IBA algorithm is evaluated in comparison with the results obtained from other authors' algorithm for a number of benchmark instances. The proposed algorithm is very effective and efficient. It can find optima for a set of test instances, and running time is less than another algorithm.

BA can be applied to many optimization problems. These results indicate that the proposed algorithm is an attractive alternative for solving the job shop scheduling problem and other optimization problems. Because BA was originally proposed for continuous optimization problems, new attempt has been made by using priority based representation to be suitable for solving discrete optimization.

References

[1] Wul C., Zhang N., Jiang J., Jinhui Yang J. and Liang Y. 2007 Improved Bacterial Foraging Algorithms and Their Applications to Job Shop Scheduling Problems, Springer-Verlag Berlin Heidelberg.

[2] Tomas K., Frantisek K. 2011 Solving job shop scheduling with the computer simulation, the International Journal of transport & logistic, ISSN 1451-107X.

[3] Cheung W. and Zhou H. 2001 Using genetic algorithms and heuristics for job shop scheduling with sequence-dependent setup times. Annals of Operations Research, Vol. 107, pp: 65–81.

[4] Sun J.U. and Yee S.R. 2003 Job shop scheduling with sequence dependent setup times to minimize makespan, International Journal of Industrial Engineering: Theory Applications, pp: 455–461.

[5] Liang S., Xiaochun C. and Yanchun L. 2010 Solving Job Shop Scheduling Problem Using Genetic

Algorithm with Penalty Function, International Journal of Intelligent Information Processing Volume 1, Number 2.

[6] Luchoomun K., Auckloo P. and Sonah B. 2014 Enhancing Performance of Genetic Algorithm for Static Job-Shop Scheduling Problems, International Refereed Journal of Engineering and Science (IRJES), Vol. 3, PP:39-49.

[7] Miguel A. S. , Joan E., Adriana G. and Federico B. 2015 A genetic algorithm for energy efficiency in job shop scheduling, The International Journal of Advanced Manufacturing Technology, pp: 1-12.

[8] Anan B., Booncharoen S. and Tiranee A. 2012 Job Shop Scheduling with the Best-so-far ABC, Engineering Applications of Artificial Intelligence 25, pp: 583–593.

[9] Tahar D.N., Yalaoui F., Amodeo L. and Chu C. 2005 An ant colony system minimizing total tardiness for hybrid job shop scheduling problem with sequence dependent setup times and release dates, In: Proceedings of the International Conference on Industrial Engineering and Systems Management, Marrakech, Morocco, pp: 469–478.

[10] Huang R. H. and Yang C. L. 2008 Ant colony system for job shop scheduling with time windows, The International Journal of Advanced Manufacturing Technology, vol. 39, pp: 151-157.

[11] Zhu R., Wang S., Zhu Z. and Yi L. 2014 an ant colony algorithm for job shop scheduling problem with tool flow, Journal of engineering manufacture.

[12] Shruti R. R. and Rajiv G. 2015 Review on development of meta-heuristic based solution for job shop scheduling problem, international Journal of engineering science & research technology, pp: 459-462.

[13] Lei D. 2008 A Pareto archive particle swarm optimization for multi-objective job shop scheduling, Computers & Industrial Engineering, pp: 960-971.

[14] Lin T.-L., Horng S.-J., Kao T.-W. Chen Y.-H., Run R.-S., Chen R.- J., et al. 2010 An efficient job-shop scheduling algorithm based on particle swarm optimization, Expert Systems with Applications,pp: 2629–2636.

[15] Lian Z., Jiao B and Gu X. 2006 A similar particle swarm optimization algorithm for job-shop scheduling to minimize makespan, Applied Mathematics and Computation, pp: 1008–1017.

[16] Pisut P. 2014 A self-tuning PSO for job-shop scheduling problems, Int. J. Operational Research, Vol. 19, pp: 96- 113.

[17] Artigues C., Buscaylet F. and Feillet D. 2005 Lower and upper bounds for the job shop scheduling problem with sequence dependent setup times, In: Proceedings of the 2nd Multidisciplinary International Conference on Scheduling: Theory and Applications, New York, USA, pp: 316–321.

[18] Jean-Paul W., Christopher J., Adele E. H. and Darrell W. L. 2003 Problem difficulty for tabu search in job shop scheduling, Artificial Intelligence, www.elsevier.com/locate/artint, pp: 189–217.

[19] Zhu Z.C., Ng K. M. and Ong H. L. 2010 A modified tabu search algorithm for cost based job shop problem, Journal of the Operational Research Society, pp: 611-619.

[20] Atabak E., Maghsud S., Seyda T. and Afshin E. 2011 A simulated annealing algorithm for the job shop cell scheduling problem with intercellular moves and reentrant parts, Computers & Industrial Engineering 61, pp: 171–178.

[21] Daniil A. Z. and Valery A. K. 2014 Job Shop Scheduling and Co-Design of Real-Time Systems with Simulated Annealing, In Proceedings of the 3rd International Conference on Operations Research and Enterprise Systems, pp: 17-26.

[22] Nakandhrakumar R. S. and Balachandar M. 2014 Implementation of Simulated Annealing Technique for Optimizing Job Shop Scheduling Problem, International Journal of Advanced Mechanical Engineering, Vol. 4, pp:169-174

[23] Najid N.M., Dauzere P. and Zaidat A. 2002 A modified simulated annealing method for flexible job shop scheduling Problem, Proceedings of the IEEE International Conference on Systems, USA, IEEE, PP: 89-94.

[24] Sha D.Y. and Cheng-Yu. 2006 A hybrid particle swarm optimization for job shop scheduling problem, Computers & Industrial Engineering, pp: 791–808.

[25] Kamrul H. S. M. and Ruhul S. and David C. 2015 Hybrid Genetic Algorithm for Solving Job Shop Scheduling Problem, IEEE Xplore.

[26] Tsung-Lieh L., Shi-Jinn H., Tzong-Wann K., Yuan-Hsin C., Ray-Shine R., Rong-Jian C., Jui-Lin L. and Hong K. 2010 An efficient job-shop scheduling algorithm based on particle swarm optimization, Expert Systems with Applications, pp: 2629–2636.

[27] Yang X. S. 2010 A New Meta heuristic Bat-Inspired Algorithm, in: Nature Inspired Cooperative Strategies for Optimization, Studies in Computational Intelligence, Springer Berlin, 284, Springer, 65-74.

[28] Cheng R., Gen M. and Tsujimura Y. (1996). A tutorial survey of job-shop scheduling problems using genetic algorithms. Computers and Industrial Engineering, pp: 983–997.

[29] Kanate P. and Anan M. 2010 Algorithm for Solving Job Shop Scheduling Problem Based on machine availability constraint, International Journal on Computer Science and Engineering(IJCSE), Vol. 02, No. 05, 1919-1925.

[30] Giffler J. and Thompson G. L. 1960 Algorithms for solving production scheduling problems, Operations Research, pp: 487–503.

[31] Fisher H. and Thompson G. L. 1963 Industrial scheduling, Englewood Cliffs, NJ: Prentice-Hall.

[32] Lawrence S. 1984 Resource constrained project scheduling: An experimental investigation of heuristic scheduling techniques, Graduate School of Industrial Administration (GSIA), Carnegie Mellon University, Pittsburgh, PA.

[33] Beasley, J. E. (1990). OR-Library: Distributing test problems by electronic mail. Journal of the Operational Research Society, pp.1069–1072.