A novel Improved Bat Algorithm for Job Shop Scheduling Problem

Hegazy Zaher
Prof. of Mathematical Statistics
Institute of Statistical studies & Research,Cairo University, Egypt

Mahmoud El-Sherbieny,
Prof. of Operations Research
Institute of Statistical studies & Research,
Cairo University, Egypt

Naglaa Ragaa
Assoc. Prof. of Operations Research,
Institute of Statistical studies & Research,
Cairo University, Egypt

Heba Sayed
Candidate of Doctoral in Operations Research
Institute of Statistical studies & Research,
Cairo University, Egypt

ABSTRACT
This paper introduces a novel improved bat algorithm for solving job shop scheduling problem reaching to the optimal. A proposed novel improved Bat Algorithm plays an important role in effective and efficient computations of function optimization for job shop scheduling problem.

In this paper, an optimization algorithm based on improving Giffler and Thompson algorithm through recognizing a non-delay schedule for starting time instead of finishing time to solve the NP-hard job shop scheduling problem.

For improving the diversity of population, enhance the quality of the solution, swap operator is used to-enhance the solution.

This paper is based on ten benchmarking problems. The results demonstrate that the proposed novel improved algorithm gives better results than the particle swarm algorithm and our previous modified algorithm in both convergence speed and accuracy.

General Terms
Bat Algorithm, Job Shop Scheduling Problem.

Keywords
Job Shop Scheduling, Makespan, Bat Algorithm, Priority based representation, Giffler and Thompson Algorithm and Swap operator.

1. INTRODUCTION
Scheduling problems are one of the most important in manufacturing planning. They are one of the most difficult problems in the area combinatorial optimization [1].

The scheduling is the process of allocating limited resources to the operations (activities) over time [2].

Scheduling problems are formulated in terms of machines and jobs. The machines represent resources and the jobs represent tasks that have to be carried out using these resources [3].

The scheduling problems are divided into several types such as Single machine problems, Multi-machine problems, Single-stage problems and Multi-stage problems.

When the problem involves Multi Stage Multi Machine Problem; it is Job Shop Scheduling Problem (JSSP) or Flow Shop Scheduling Problem (FSSP) [4].

In FSSP, there are m machines in the series. Each job has to be processed on each one of the m machines. All jobs have to follow the same route (i.e., they have to be processed first on machine 1, then machine 2, etc.) [5].

In JSSP, each job has its own predetermined route to follow.

The problem is to find a schedule to minimize the makespan (Cmax) that is the time required to complete all jobs.

In a JSSP, any feasible solution is called a feasible schedule. Feasible schedules for a JSSP can be classified into three types: semi-active, active and non-delay [6, 7].

Meta-heuristic methods are used to solve JSSP, such as Genetic algorithms (GA), Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Tabu Search (TS), Simulated Annealing (SA), hybrid PSO, hyprid GA and hybrid swarm intelligence algorithmis as in [7].

Bat Algorithm (BA) was introduced by Yang (2010). It is a new meta-heuristic optimization algorithm observing and searching for the prey of the bats.

In this paper, a novel Improved Bat Algorithm (IBA) is applied to solve the JSSP. The optimal JSSP solution should be a non delay schedule, thus, improved Giffler and Thompson's heuristic is applied to decode a bat position into a schedule.

Swap operator: it is choosing two different positions from a job permutation randomly and swap them [8].

This paper is structured as follows. Section 2 presents "Methodology" which includes four subsections as follows: "Job Shop Scheduling Problem", "Bat Algorithm", "Priority-based representation" and "Improved Giffler and Thompson Algorithm". Section 3, considers "Proposed Improved Bat Algorithm for Job Shop Scheduling Problem" and it is also includes two subsections "Improved Bat Algorithm" and "Steps of Proposed Model". Section 4 shows "Computational results" which the proposed improved bat algorithm for JSSP is tested on Fisher and Thompson (1963) and Lawrence (1984) test problems. Finally, conclusion and further works are given in Section 5.

2. METHODOLOGY
2.1 Job Shop Scheduling Problem
JSSP is defined as follows: - There is a machine set $M = \{M_1, M_2, \ldots, M_m\}$ and a job set $J = \{J_1, J_2, \ldots, J_n\}$. Each job J must go through m machines to complete its work. One job consists of a set of operations, and the operation order for the machines is predetermined. Each operation uses one of them machines to complete one job's work for a fixed time interval [9].

The main purpose of JSSP is commonly used to find the best machine schedule for servicing all jobs in order to optimize

either single criterion/objective or multi scheduling objectives. They are also known as job shop performance measures such as the makespan minimization (C_{max}) or mean flow time or the mean tardiness or earliness etc. The makespan is the maximum total completion time of the final operation in the schedule of $n \times m$ operations.

In general, the constraints used in job shop scheduling are:

- Each job must be processed by each machine in a certain order (precedence constraints)
- Each machine can only process one job at a time
- Each job can only be processed by one machine at a time
- No preemption is allowed, or once a job has started processing it cannot be interrupted.

The general job shop scheduling mathematical model as presented as in [10]. The detail of machine availability constraint and variables 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 the machine i,

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 the machine i,

Let $P_{i,j}$ be processing time of job j that is performed on the 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 the job shop scheduling problem.

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

St.

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

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

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

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

To make sure that the next step on machine h of job j starts after the finish time of the step on the machine i of job j, equation 2 is employed. Next, equation 3 ensures that Cmax 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 nonnegative.

2.2 Bat Algorithm

BA is an evolutionary algorithm introduced by Yang in 2010. The advantage of BA is that it can provide very quick convergence at a very initial stage by switching from exploration to exploitation. 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 any way. Moreover, PSO is the special case of the BA under appropriate simplifications. Steps of BA are as follows [11]:

A. Initialization of Bat Population

Initial population is randomly generated from real valued vectors with dimension d and a number of bats and, by taking into account lower and upper boundaries.

$$x_{ij} = x_{minj} + rand(0,1)\left(x_{maxj} - x_{\min j}\right) \tag{1}$$

Where $i=1, 2, \ldots n$, $j=1, 2, \ldots. d$, x_{minj} and x_{maxj} are lower and upper boundaries for dimension j respectively.

B. Update Process of Frequency, Velocity and Solution

The frequency factor (Q_i) controls the step size of a solution in BA. This factor is assigned to a random value for each bat (solution) between upper and lower boundaries [Q_{min}, Q_{max}]. Velocity (v_i) of a solution is proportional to frequency and new solution (x_i) depends on its new velocity.

$$Q_i = Q_{min} + (Q_{max} - Q_{\min}) * \beta \tag{2}$$

$$v_i^t = v_i^{t-1} + (x_i^t - x_{best}) * Q_i \tag{3}$$

$$x_i^t = x_i^{t-1} + v_i^t \tag{4}$$

Where β ϵ [0, 1] indicates randomly generated number, x_{best} represents current global best solutions (exploration), which is located after comparing all the solutions among all the n bats.

The pulse emission rate is denoted by the symbol r_i, and ∈ [0, 1]. In every iteration, a random number is generated and compared with r_i. If the random number is greater than i, a local search part of the algorithm (exploitation), one solution is selected among the selected best solutions and random walk is applied.

$$x_{new} = x_{old} + \varepsilon \overline{A^t} \tag{5}$$

Where$\overline{A^t}$, is the average loudness of all bats, ε ϵ [0, 1] is a random number and represents the direction and intensity of random-walk.

C. Update Process of Loudness and Pulse Emission Rate

Loudness (A_i) and pulse emission rate (r_i) must be updated only when the global near best solution is updated and the randomly generated number is smaller than A_i. Loudness usually decreases and pulse emission rate increases. Loudness (A_i) and pulse emission rate (r_i) are updated by the following equations:

$$A_i^{t+1} = \alpha * A_i^t \tag{6}$$

$$r_i^{t+1} = r_i^0[1 - e^{-\gamma t}] \tag{7}$$

2.3 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 a n job m machine problem, we can represent the bat k position by an m n matrix, i.e.

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

Wherex_{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.

2.4 Improved Giffler and Thompson Algorithm

A bat position can be mapped (or decoded) into a non delay schedule using improved Giffler and Thompson's heuristic. An optimization algorithm based on improving Giffler and

Thompson algorithm through recognizing a non-delay schedule for starting time instead of finishing time is applied to solve the NP-hard job shop scheduling problem. The improved Giffler and Thompson (G&T) algorithm is described as follows [15]:

Notation:

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

$T_{(i,\ i)}$: 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 $(i, j) \in U$ can be started.

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

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

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

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

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

Step 3:

- Identify the operation set $(i', j') \in U$ such that (i', j') requires machine m^*, and $s_{(i',j')} == s^*$.
- Choose (i, j) from the operation set identified in (1) with the largest priority.
- 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.

3. PROPOSED IMPROVED BAT ALGORITHM for JOB SHOP SCHEDULING PROBLEM

JSSP is a combinatorial problem, and its solution space is discrete. However, the encoding scheme for Improved Bat Algorithm (IBA) is continuous, so it cannot be applied to solve JSSP directly. In order to apply IBA to JSSP, a direct mapping relationship between the job sequence and the vector of individuals in IBA must be constructed as a discrete one. To do so, a priority based representation, then, the improved Giffler and Thompson algorithm through recognizing a non-delay schedule for starting time instead of finishing time and then, the swap operator is used to enhance bat solutions as shown in Appendix 1. IBA is an effective way to obtain a more effective solution.

3.1 Improved Bat Algorithm

BA is a powerful algorithm at exploitation, but it has some insufficiency at exploration [12]. Thus, it can easily get trapped in local minima on most of the multimodal test functions. In order to overcome this problem of standard BA, there is modification to improve exploration and exploitation capability of BA as presented in [13]. This modification is Inertia Weigh Factor Modification.

The update process of the velocities and positions of bats has some similarity to the procedure of standard Particle Swarm Optimization (PSO).

In standard BA exploration and exploitation are controlled by pulse emission rate r, and this factor increases as iteration proceeds, thus an algorithm gradually loses exploitation capability as iteration proceeds. To avoid this problem, exploitation capability of BA is improved by inserting linear decreasing inertia weight factor [14].

When inertia weight factor is decreasing linearly, previous velocity is gradually decreasing. Thus the exploitation rate of BA gradually increases as the iterations proceed.& we modified inertia weight factor with the following formula:

$$w_{iter} = w_{max} - (w_{min} * \left(\frac{iter_{max} - iter}{iter_{max}}\right)^{n}) \quad (8)$$

Where $iter$ is current iteration value, $iter_{max}$ is maximum iteration number; w_{max} and w_{min} are maximum and minimum inertia weight factor respectively.

Thus, the velocity equation becomes:

$$v_i^t = w_{iter} * v_i^{t-1} + (x_i^t - x_{best}) * Q_i \quad (9)$$

3.2 Steps of Proposed Model

This section describes how to combine between IBA and improved G&T algorithm to solve JSSP as follows:

Step 1: Initialization
Generate bat population in continuous space x_i (i = 1, 2, ..., p), define pulse frequency f_i and velocity v_i at x_i and Initialize pulse rates r_i and the loudness A_i.

Step 2: Representation of individuals
Representation of bats in IBA for JSSP using priority based representation.

Step 3: Evaluation
Evaluate the value of the fitness function (makespan) using improved Giffler and Thompson algorithm.

Step4 : Update
Generate new solutions by adjusting frequency and updating velocities and locations/solutions using Eqs. (2), (9) and (4).

Step 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.(5).

Step 6: Repeat step 3.

Step 7: Update Process of Loudness and Pulse Emission Rate:
Loudness (A_i) and pulse emission rate (r_i) must be updated using Eqs. (6) and (7) only if the global best solution is updated and the randomly generated number is smaller than A_i

Step 8: Proposed Swap Operator:
for i = 1: p

Execute swap operation based on p$best\ x$ by using Pseudo code of swap operator.

For improving the diversity of population, enhance the quality of the solution, one of operations has to be used the

application methods of the diversity-enhanced solutions .This operation is swapping.

Swap is to choose two different positions of an operation permutation sequence randomly and swap them.

For example, when two positions of an operation sequence in an individual need to exchange the order, the "swap" operator has to be used. Compare the makespan before being exchanged and after being exchanged, if the makespan after being exchanged is better, then the new permutation operation of this individual of the current solution will be updated.

```
k=1:m, // for machine 1 to machine m
rand ϵ U(0,1)
if rand<=Ws,
L1=randi(n,1,1)  // an integer random number between 1 to n
L2=randi(n,1,1) // an integer random number between 1 to n
  L1==L2,
   L2=n - L1

J1=S(k,L1,i) // the location of J1 in pbest i
J2=S(k,L2,i)  // the location of J2 in pbest i
pbest(k,L1,i)=J2;
```

Fig 1: Pseudo code of proposed swap operator

Step 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.

4. COMPUTATIONAL RESULTS

In the experiment, ten instances of the benchmark in job shop scheduling data set with various sizes are used to test the behavior of the convergence, and accuracy of the proposed method. The results compared with the other methods in the literature show that the proposed scheme increases more the convergence and the accuracy than BA and particle swarm optimization.

The IBA is tested on problems (FT06, FT10, and FT20) [16], (LA01 to LA07) [17]. These problems are available on the OR-Library web site [18]. IBA is tested with 30 independent runs; the number of individuals (bat) in the population is fixed to 30. Maximum iterations for the priority-based IBA is 500 for each run. Boundaries of inertia factors w_{max} and w_{min} are fixed to 0.9 and 0.2 respectively. Q_{min} is 0 and Q_{max} is 1 while α and γ are 0.9 for IBA. The proposed algorithm is compared with the priority-based PSO [19] and priority-based PBA [7]. 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), best solution, average and Relative Percentage Error (RPE) for each method.

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

The results show that the priority based IBA is much better than the priority based PSO and PBA because IBA is based on improve G&T algorithm. The main difference between improved G&T algorithm and G&T algorithm is in the Step 2.

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

Table 1. The comparison between PSO, PBA and IBA

	problem	Size (n*m)	Optimal	PSO-priority based			PBA-priority based			IBA-priority based		
				best	Average	RPE	best	Average	RPE	best	Average	RPE
1	Ft06	6*6	55	55	58.9	0	55	56.8	0	55	56.93333	0
2	Ft10	10*10	930	1007	1086	8.27957	1004	1076.567	7.956989	991	1012.6	6.55914
3	Ft20	20*5	1165	1242	1296	6.609442	1203	1283.7	3.261803	1177	1188.533	1.030043
4	La01	10*5	666	681	705	2.252252	666	695.9	0	666	666	0
5	La02	10*5	655	694	729.7	5.954198	672	696.9667	2.59542	668	677.5	1.984733
6	La03	10*5	597	633	657.5	6.030151	621	633.4667	4.020101	613	622.6	2.680067
7	La04	10*5	590	611	648.1	3.559322	610	633.3	3.389831	611	611	3.559322
8	La05	10*5	593	593	601.1	0	593	599.8	0	593	593	0
9	La06	15*5	926	926	940.2	0	926	938.5	0	926	926	0
10	La07	15*5	890	890	940.1	0	890	934.9333	0	890	890	0

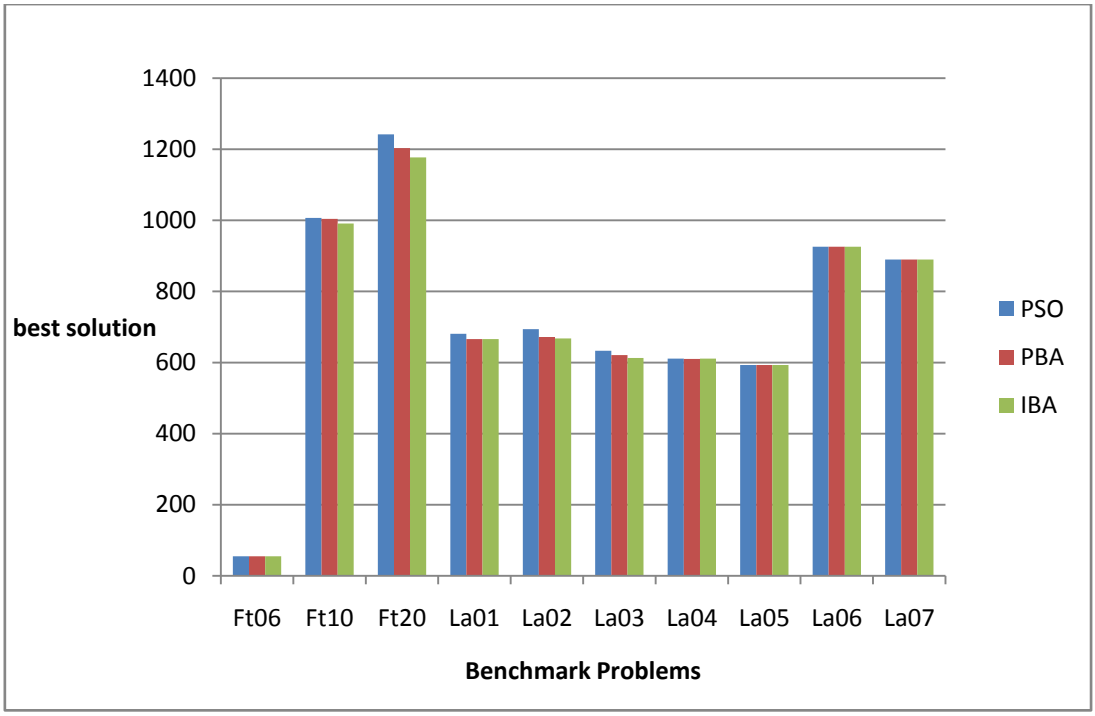


Fig 2: Comparison between PSO, PBA and IBA using best solution

In Fig. 3, The comparison between PSO, PBA and IBA is based on Relative Percent Error (RPE). The three algorithms are able to find the best known solutions for the problems (Ft06, La05, La 06 and La07) of PSO, PBA and IBA, and therefore RPE is zero. But RPE of the problem La01 is zero for PBA and IBA. For the problems (Ft10, Ft20, La02, La03, and La04), that RPE of IBA is better than RPE of PSO and PBA.

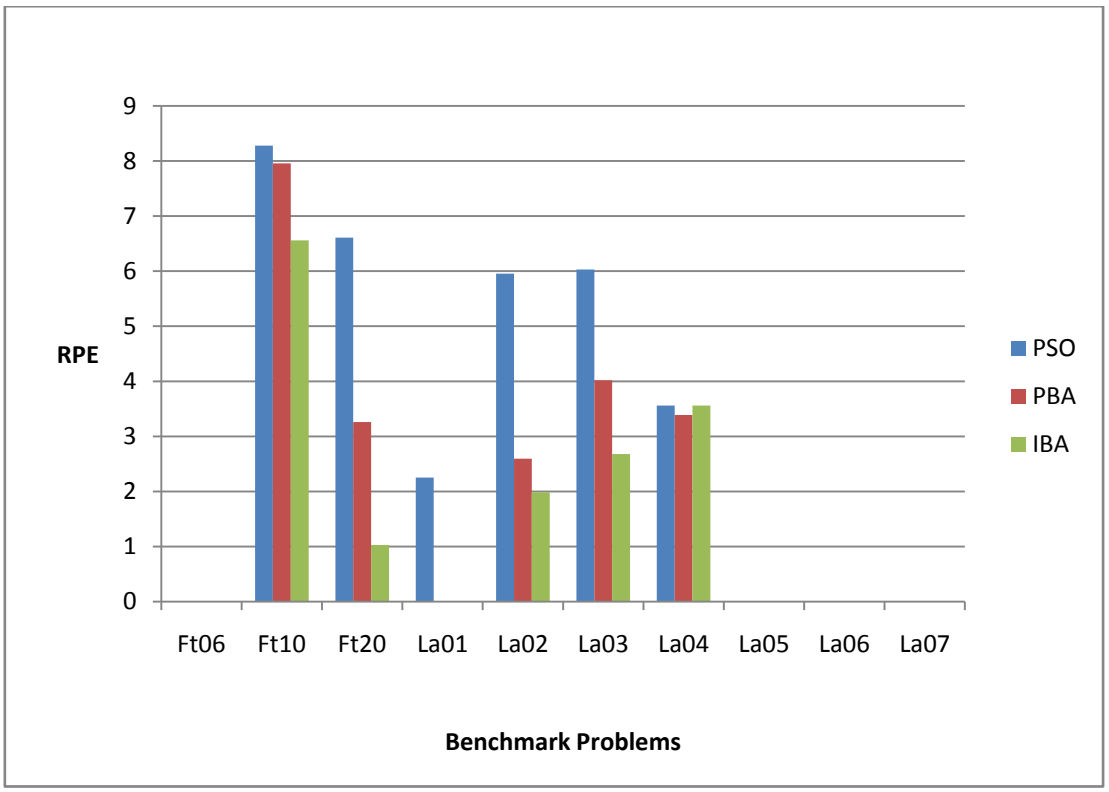


Fig 3: Comparison between PSO, PBA and IBA using RPE

5. CONCOLUSION

The novel improved bat algorithm is proposed to minimize makespan of the JSSP based on G&T algorithm by checking a non delay schedule for starting time instead of finishing time.

Although active schedule generation can always find optimal solution, but it is not used frequently in the real world practice due two major reasons: The earliest finishing time is difficult to be predetermined exactly, and so the finish time of an operation due to real world delays (Machine breaks, worker failures, transport delays etc.). The search space in the active schedule generation is much greater than in the case of non-Delay schedule.

The performance of IBA is evaluated by comparing the results obtained from all previous authors' algorithms (including us) for a number of benchmark instances. it is clear

that the proposed novel improved algorithm is very effective and efficient. It can find optima for a set of test instances, and running time is less than other algorithm.

We will attempt to apply BA to other shop scheduling problems with multiple objectives in future research. Other possible topics for further study include modification of the bat position, bat movement, and bat velocity representation.

6. REFERENCES

[1] Wu1 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] Peter B. 2006 Scheduling Algorithms, Springer-Verlag Berlin Heidelberg.

[3] Frits C. R. S. 2007 Interval Scheduling, 3rd Multidisciplinary International Conference on Scheduling: Theory and Application, Paris, France.

[4] Shiva-Kumar B. L. and Amudha T. 2013 Enhanced Bacterial Foraging Algorithm For Permutation Flow Shop Scheduling Problems, Asian Research Publishing Network (ARPN).

[5] Jen S. C. and Jin S.Y. 2006 Model formulation for the machine scheduling problem with limited waiting time constraint, Journal of Information & Optimization Sciences.

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

[7] Hegazy Z., Mahmoud El-S., Naglaa R. S., Heba S. 2017 Bat Algorithm for Job Shop Scheduling Problem, Journal of Multidisciplinary Engineering Science and Technology (JMEST) ISSN: 2458-9403, Vol. 4 Issue 2, pp: 6758-6763.

[8] Qifang L., Yongquan Z., Jian X.,Mingzhi M., and Liangliang L. 2014 Discrete Bat Algorithm for Optimal Problem of Permutation Flow Shop Scheduling, Scientific World Journal , 15 pages.

[9] Gonçalves J. F., Mendes J. J. d. M. and Resende M. G. C. 2005 A hybrid genetic algorithm for the job shop scheduling problem, European Journal of Operational Research, pp: 77–95.

[10] 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.

[11] 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.

[12] Wang G. and Guo L. 2013 A Novel Hybrid Bat Algorithm with Harmony Search for Global Numerical Optimization, Journal of Applied Mathematics, vol. 2013, pp. 21.

[13] Selim Y. and Ecris U. K. 2013 Improved Bat Algorithm (IBA) on Continuous Optimization Problems, Lecture Notes on Software Engineering, Vol. 1, No. 3.

[14] Nickabadi A., Ebadzadeh M. and Safabakhsh R. 2011 A novel particle swarm optimization algorithm with adaptive inertia weight, Applied Soft Computing, vol. 11, pp. 3658-3670.

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

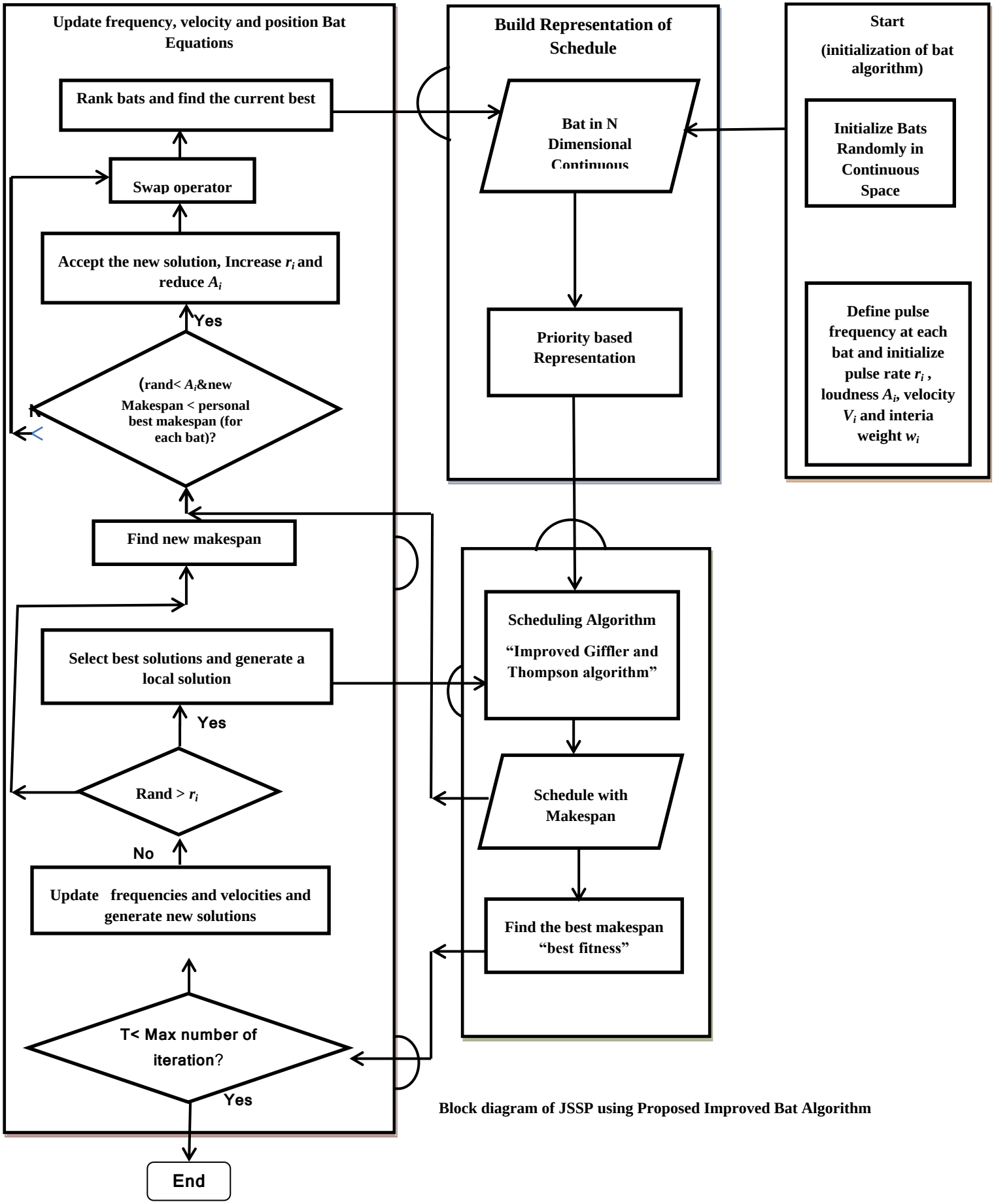
[16] Fisher H. and Thompson G. L. 1963 Industrial scheduling, Englewood Cliffs, NJ: Prentice-Hall.

[17] 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.

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

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

7. APPENDIX 1



Block diagram of JSSP using Proposed Improved Bat Algorithm