AN IMPROVED BIO-INSPIRED METHODOLOGY TO SOLVE FLOW SHOP SCHEDULING PROBLEMS

Divya. P
Amudha. T
Narendhar. S

ABSTRACT

Optimization is a mathematical discipline, which involves the operation of finding minima and maxima of functions. Scheduling refers to resource allocation process, which is used to complete activities in an efficient manner. This paper represents the efficiency of Modified Bacterial Foraging Optimization algorithm. In this research work Bacterial Foraging Optimization was hybridized with Ant Colony Optimization and a new technique for solving Flow Shop Scheduling Problem was proposed. The proposed Modified Bacterial Foraging Optimization was implemented to solve the Benchmark instances of Flow Shop Scheduling Problem. Results have shown that the proposed Modified Bacterial Foraging Optimization has outperformed Bacterial Foraging Optimization in arriving at improved best makespan for various test instances of Flow Shop Scheduling Problem.

Keywords: Ant Colony Optimization (ACO), Bacterial Foraging Optimization (BFO), Flow Shop Scheduling Problem (FSSP), Modified Bacterial Foraging Optimization (MBFO)

I. INTRODUCTION

Scheduling is allocation of resources to activities over time and cost. Scheduling classifications includes Job Shop Scheduling Problem (JSSP), Flow Shop Scheduling Problem (FSSP), Open Shop Scheduling Problem (OSSP), Mixed Shop Scheduling Problem (MSSP), Group Shop Scheduling Problem (GSSP) etc. Scheduling constraints are (i) Each machine can only process one job at a time. (ii) Each job can only be processed by one machine at any time. (iii) Once a machine has started processing a job, it will continue running on that job until the job is finished.

Nature-Inspired computing paves way to develop new computing technique which is based on nature behaviour in solving complex problems. Some popular Nature-Inspired Metaheuristics are Evolution Strategy (ES), Evolutionary Programming (EP), Genetic Algorithm (GA), Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO) [8], Differential Evolution (DE), Bacterial Foraging Optimization (BFO), Honey Bee (HB), Harmony Search (HS), Artificial Bee Colony (ABC). Bio-Inspired computing is the subset of Nature-Inspired computing.

A. ANT COLONY OPTIMIZATION

ACO algorithm first proposed by M. Dorigo, in 1992 [29]. It is a metaheuristic in which a colony of ants capable of finding shortest path from their nest to food sources using pheromone trials. The probability that the ants coming later choose the path is proportional to the amount of pheromone on the path, previously deposited by other ants.

Ants want to find food, so they set off from their nest and arrive at a decision point at which they have to decide which path to go on, for there are three different paths. Since they have no clue about which is the best choice, they choose the path just randomly, and on average the numbers of ants on every path are the same. Suppose that all ants walk at the same speed and deposit the same amount of pheromone. Since the middle path is the shortest one, ants following this path reach the food point first. Therefore more ants will complete their tour through the middle path in the same period of time, and more pheromone will be deposited in this road correspondingly. When ants return to their nest after they find the food; since there is more pheromone in the middle path, ants will prefer in probability to choose the middle path. This in turn increases the number of ants choosing the middle and shortest path. This is a positive feedback effect with which very soon all ants will follow the shortest path.

B. BACTERIAL FORAGING OPTIMIZATION

BFO was introduced by Kevin M. Passino in 2000 for distributed optimization problems [8]. Bacterial Foraging Optimization (BFO) algorithm is a novel evolutionary computation algorithm proposed based on the foraging behavior of Escherichia coli (E. coli) bacteria living in human intestine. The BFO algorithm is a biologically inspired computing technique which is based on mimicking the foraging behavior of E. coli bacteria [15].

Natural selection tends to eliminate animals with poor foraging strategies and favors the propagation of genes of those animals that have successful foraging strategies, since they are more likely to enjoy reproductive success. After many generations, poor foraging strategies are either eliminated or shaped into good ones. This activity of foraging led researchers to use it as optimization process.

Framework for BFO algorithm

- Input the bacterial foraging parameters and independent variable, then specify lower and upper limits of the variables and initiate the elimination-dispersal steps, reproduction and chemotactic.
- Generate the positions of the independent variable randomly for a population of bacteria. Evaluate the objective value of each bacterium.
- Modify the position of the variables for all the bacteria using the tumbling or swimming process.
- Perform reproduction and elimination operation.
- If the maximum number of chemotactic, reproduction and elimination-dispersal steps is reached, then output the variable corresponding to the overall best bacterium; Otherwise, repeat the process by modifying the position of the variables for all the bacteria using the tumbling /swimming process .

C. FLOW SHOP SCHEDULING PROBLEM

Johnson's Rule (Johnson, 1954) has been the basis of many FSSP heuristics. Palmer (1965) first proposed a heuristic for the FSSP to minimize makespan. FSSP are defined by a set of n jobs, where each job has to be processed in an identical order on a given number of m machines. Each machine can process only one job at a time. The parameters $t_{ij}, 1 \leq i \leq n, 1 \leq j \leq m$, denote The processing time of job i on machine j [1]. The FSSP is a set of jobs that flow through multiple stages in the same order as shown in the figure.

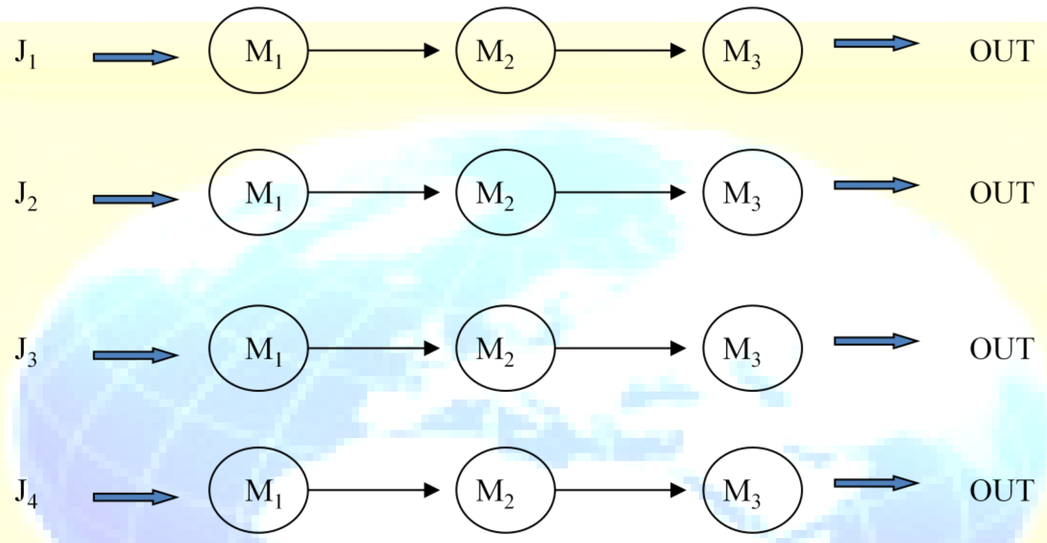


Figure I: Flow of FSSP

For continuous FSSP the processing of each job has to be continuous, which means that there must not be any waiting times between the processing of any consecutive tasks of this job. To allow processing of a job without interruption on all machines, the order in which the jobs are processed on a machine is the same for all machines (assuming non-zero processing times). If a job does not have to be processed on some machine (zero processing time on this machine), passing could occur without violating continuous processing.

Constraints

The machines in a flowshop are capable of processing at most one job at a time, and each job can be processed on at most one machine at any time. The n-jobs are independent, simultaneously available at time zero, and the machine sequences of all jobs are the same. Each job has a known and finite processing time on each machine, and the processing times are independent of the order in which operations are carried out.

In this research work, BFO algorithm was hybridized with ACO and a new Modified Bacterial Foraging Optimization (MBFO) algorithm was proposed. Both BFO and MBFO algorithm were applied to Carlier (car) and randomly created instances (RND). The results obtained by MBFO algorithm were compared with BFO, Lower Bound and Upper Bound of Benchmark problems.

II. RELATED WORKS

Samia kouki, Mohamed Jemni, Talel Ladhari (2011) have proposed a paper about Solving the Permutation Flow Shop Problem with Makespan Criterion using Grids. The optimization of scheduling problems is based on different criteria to optimize. One of the most important criteria is the minimization of completion time of the last task on the last machine called makespan. They presented a parallel algorithm for solving the permutation flow shop problem. This is used to minimizing the total makespan of the tasks by using Branch and Bound method to find optimal solutions

According to Hela Boukef, Mohamed Benrejeb and Pierre Borne [2006-2007] have proposed a new genetic algorithm coding is proposed in this paper to solve flow-shop scheduling problems. To show the efficiency of the considered approach with minimization of different costs related to each problem as a scope. Multi-objective optimization is thus, used and its performances proved. The principle scope of this method, based on natural selection mechanism, is the improvement of robustness and balance between cost and performance [12].

AndreasFink, StefanVoß (2001) have proposed a paper about Solving the Continuous Flow Shop Scheduling Problem by Metaheuristics. This problem is used to find a permutation of jobs to be processed sequentially on a number of machines under the restriction that the processing of each job has to be continuous with respect to the objective of minimizing the total processing time (flow-time). i.e., once the processing of a job begins, there must not be any waiting times between the processing of any consecutive tasks of this job [1].

Jing Dang, Anthony Brabazon, Michael O'Neill, and David Edition (2008) have proposed a paper about Bacterial Foraging Optimization (BFO) algorithm. This is a biologically inspired computation technique which is based on mimicking the foraging behavior of E. coli bacteria. During the lifetime of E.coli bacteria, they undergo different stages such as chemotaxis, reproduction and elimination-dispersal. BFO algorithm can be constructed and applied to solve various real world problems, in a number of application domains. Kim suggested that the BFO could be applied to find solutions for difficult engineering design problems [15].

According to S. Subramanian and S. Padma (2011), the selection behaviour of bacteria tends to eliminate poor foraging strategies and improve successful foraging strategies. The E.coli bacterium has a control system that enables it to search for food and try to avoid noxious substances. BFO is used to minimizing cost and improves the efficiency simultaneously by using a multi objective based bacterial foraging algorithm [25].

Jun Zhang, Xiaomin Hu, X.Tan, J.H Zhong and Q. Huang (2006) presented an investigation into the use of an Ant Colony Optimization (ACO) to optimize the JSSP. ACO is extensively used to solve NP-Hard Combinatorial Optimization problems. Its original model is based on the foraging behaviour of real ants who find an approximately shortest way to the food by detecting the density of pheromone deposited on the route. The main characteristics of ACO are positive feedback, distributed computation, robustness and the use of a constructive greedy heuristic [16].

According to Katie Kinzler (2008), there are wide varieties of mechanisms used by ants to communicate as well as a variety of reasons for communication. Some of these forms of communication involve stroking, gasping, antenna movements, and streaking of chemicals. Chemicals are known as pheromones, which is a major form of communications used by ants. When foraging for food, worker ants will choose to lay pheromones depending on the quality and volume of food found at a source. If they do lay a pheromone trail, they adjust the quantity based on how profitable the food source is. Therefore, a strong pheromone path is created when a profitable food source is found [17].

III. PROPOSED METHODOLOGY FOR FLOW SHOP SCHEDULING PROBLEMS

The objectives of this research paper are

- To propose and implement Modified Bacterial Foraging Optimization (MBFO) Algorithm to solve FSSP.

- MBFO is to find a schedule that minimizes the makespan of the jobs.
- To examine the efficiency of MBFO in solving benchmark instances of FSSP.
- To analyze and compare the performance of the proposed MBFO with BFO in solving FSSP.

A. *Modified Bacterial Foraging Optimization (MBFO)*

The behavior of ant system is included in tumble part of BFO algorithm, to make it as a MBFO. Each ant builds a tour by repeatedly applying a stochastic greedy rule, which is called the state transition rule.

$$s = \begin{cases} \arg max_{u \in J(r)}\{[\tau(r,u)].[\eta(r,u)^{\beta}]\}, & if\ q \le q_0 (exploitation) \\ S, & otherwise\ (biased\ exploitation) \end{cases} \rightarrow (1)$$

(r, u) represents an edge between point r and u, and τ(r, u) stands for the pheromone on edge (r, u). η(r, u) is the desirability of edge (r, u), which is usually defined as the inverse of the length of edge (r, u). q is a random number uniformly distributed in [0, 1], q_0 is a user-defined parameter with ($0 \le q_0 \le 1$), β is the parameter controlling the relative importance of the desirability. J (r) is the set of edges available at decision point r. S is a random variable selected according to the probability distribution given below.

$$P(r,s) = \begin{cases} \frac{[\tau(r,s)].[\eta(r,s)^{\beta}]}{\sum_{u \in J(r)}[\tau(r,u)].[\eta(r,u)^{\beta}]}, & if\ s \in J\ (r) \\ 0, & otherwise \end{cases} \rightarrow (2)$$

The selection strategy used above is also called 'roulette wheel' selection since its mechanism is a simulation of the operation of a roulette wheel [13].

While ant goes for a search it will drop a certain amount of pheromone. It is a continuous process, but we can regard it as a discrete release by some rules. There are two kinds of pheromone update strategies, called local updating rule and the global updating rule.

Local updating rule

When ant constructing its tour, an it will modify the amount of pheromone on the passed edges by applying the local updating rule.

$$\tau(r,s) \leftarrow (1-\rho).\tau(r,s) + \rho.\tau o \rightarrow (3)$$

where ρ is the coefficient representing pheromone evaporation (note:$0< \rho < 1$).

Global updating rule

Once all ants have arrived at their destination, the amount of pheromone on the edge is modified again by applying the global updating rule.

$$\tau(r,s) \leftarrow (1-\alpha).\tau(r,s) + \alpha.\Delta\tau(r,s) \rightarrow \quad (4)$$

Where

$$\Delta\tau(r,s)\begin{cases}(L_{gb})^{-1}, & if\ (r,s)\epsilon global-best-tour \\ 0\,, & otherwise\end{cases} \rightarrow \quad (5)$$

Here $0<\alpha<1$ is the pheromone decay parameter, and L_{gb} is the length of the globally best tour from the beginning of the trial. Δτ(r; s) is the pheromone addition on edge (r, s). We can see that only the ant that finds the global best tour can achieve the pheromone increase [13].

In BFO, the objective is to find the minimum of J(θ),$\theta \in R^D$, where we do not have the gradient information ∇J(θ). Suppose θ is the position of the bacterium and J(θ) represents a nutrient profile, i.e.,J(θ) < 0, J(θ)=0 and J(θ)> 0 represent the presence of nutrients, a neutral medium and noxious substances respectively. The bacterium will try to move towards increasing concentrations of nutrients (i.e. find lower values of J), search for ways out of neutral media and avoid noxious substances (away from positions where J > 0). It implements a type of biased random walk.

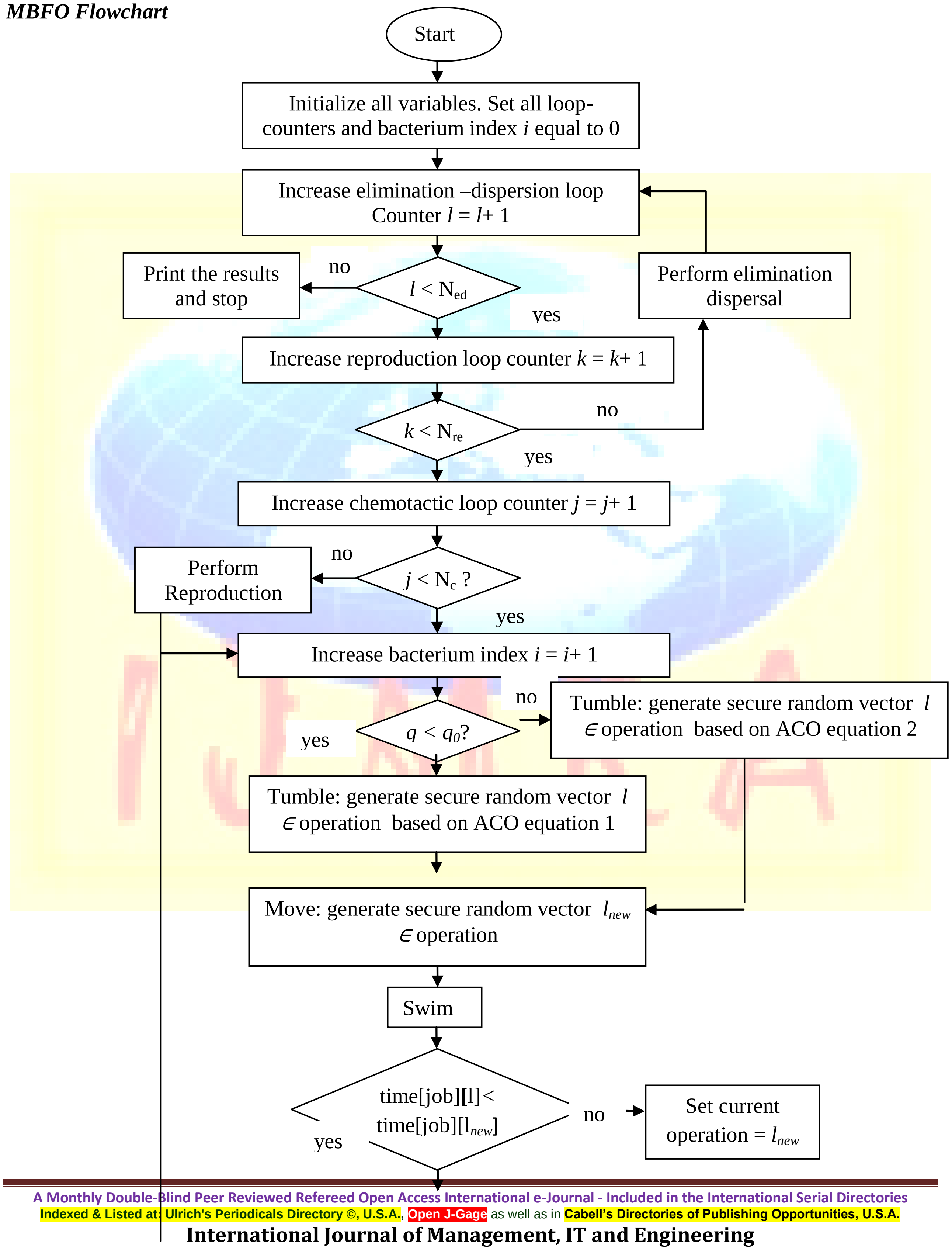
The mathematical swarming (cell-cell signalling) function can be represented by:

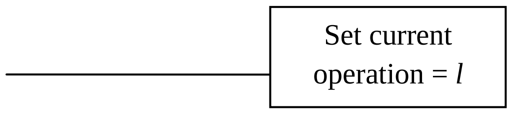
$$J_{cc}\,(\theta^i,\theta) = \begin{cases} -M\left(\sum_{k=1}^{s} e^{-w_a||\theta^i-\theta^k||^2} - \sum_{k=1}^{s} e^{-w_r||\theta^i-\theta^k||^2}\right) & With\ swarming \\ 0 & No\ swarming \end{cases} \rightarrow \quad (6)$$

Where ‖.‖ is the Euclidean norm, W_a and W_r are measures of the width of the attractant and repellent signals respectively, M measures the magnitude of the cell-cell signaling effect [12].

The above State Transition rule of ant in ACO is included in the tumble. MBFO methodology is implemented with no swarming effect (ie) $j_{cc}=0$ [15]. Here time is considered as cost. During the lifetime of E-Coli bacteria they undergo different stages such as Chemotactics, Reproduction and Elimination-Dispersal.When compared with ACO and BFO, MBFO achieves high level of SHA1PRNG algorithm incase of reproduction, elimination-dispersal.

MBFO Flowchart


Start
Initialize all variables. Set all loop-counters and bacterium index *i* equal to 0
Increase elimination –dispersion loop Counter *l* = *l*+ 1
Print the results and stop
no
l < N$_{ed}$
yes
Perform elimination dispersal
Increase reproduction loop counter *k* = *k*+ 1
k < N$_{re}$
no
yes
Increase chemotactic loop counter *j* = *j*+ 1
Perform Reproduction
no
j < N$_c$?
yes
Increase bacterium index *i* = *i*+ 1
no
Tumble: generate secure random vector *l* ∈ operation based on ACO equation 2
yes
$q < q_0$?
Tumble: generate secure random vector *l* ∈ operation based on ACO equation 1
Move: generate secure random vector l_{new} ∈ operation
Swim
time[job][l]< time[job][l$_{new}$]
no
Set current operation = l_{new}
yes



The parameters are described below in the Table I

Table I: Description of Parameters

Parameter Name	**Description**
C(i)	Step size
i	Bacterium number
j	Counter for chemotactic step
k	Counter for reproduction step
l	Counter for elimination- dispersal step
N_c	Maximum number chemotactic steps
N_{ed}	Number of elimination- dispersal events
N_{re}	Maximum number of reproduction steps

IV. COMPARISON OF RESULTS AND DISCUSSION

This paper discusses and compares the result of the implementation of ACO, BFO and proposed MBFO algorithm in solving the Benchmark instances of FSSP.

Carlier (car) benchmark problems [30] and randomly created instance (RND) for FSSP were solved in this research work. FSSP Benchmark instances were taken from Operations Research (OR) Library to test the efficiency of proposed MBFO. The proposed MBFO algorithm gave feasible solution for most runs for the constant values **ρ=*0.1, β=1.0, α=0.1, q0=0.8, τ=0.5*.** The result obtained by proposed MBFO algorithm was compared with BFO algorithm. The MBFO algorithm gave a minimum makespan, when compared with the makespan obtained by BFO

A. Comparison Results for Carlier Instances

The optimal solution obtained from proposed MBFO algorithm, BFO algorithm were compared with Lower Bound (LB), Upper Bound (UB) [22] of Carlier Instances are shown in Table II. The figure II shows the graphical representation of Table II.

Table II: Comparison Results of BFO & MBFO for Car Instances

INSTANCE	SIZE	LB	UB	BFO	MBFO
Car 1	11 * 5	7038	7817	7453	**7287**
Car 2	13 * 4	7166	7940	8051	**7643**
Car 3	12 * 5	7312	7779	**7901**	7932
Car 4	14 * 4	8003	8679	8708	**8348**
Car 5	10 * 6	7720	8773	8095	**8369**
Car 6	8 *9	8505	10211	9068	**9659**
Car 7	7 * 7	6590	7043	6869	**6942**
Car 8	8 * 8	8366	9696	8703	**9316**

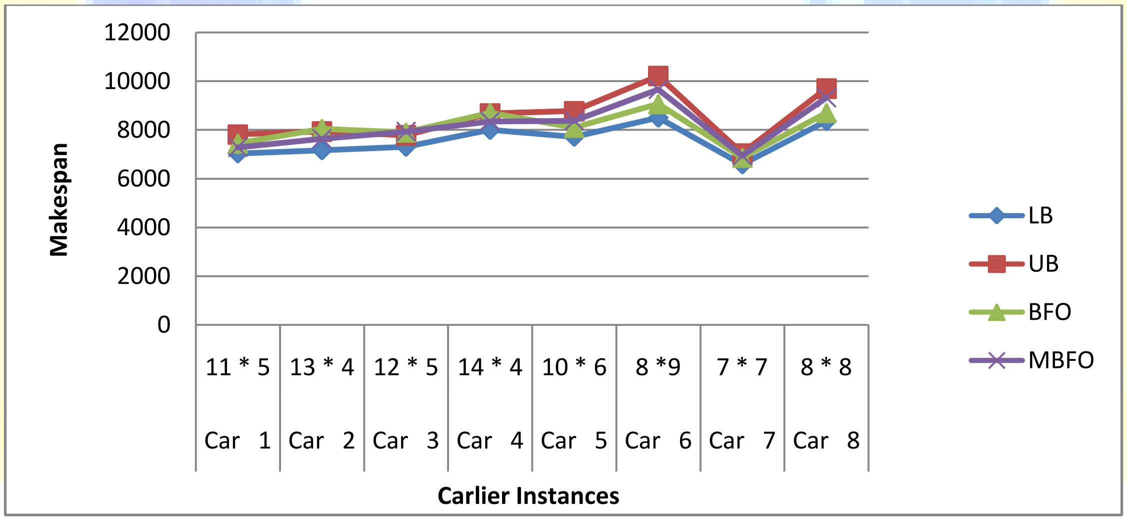


Figure II: Graphical representation of results for Carlier Instances

B. Comparison Results for RND Variable size Instances

The optimal solution obtained from proposed MBFO algorithm is compared with optimal solution obtained from BFO algorithm in solving FSSP for randomly created instances of different sizes are shown in Table III. The figure III shows the graphical representation of Table III.

Table III: Comparison Results of BFO & MBFO for RND Variable Size Instances

INSTANCE	SIZE	BFO	MBFO
RND 5	5 * 5	463	**461**
RND 10	10 * 10	1050	**1046**
RND 15	15 * 15	**1670**	1672
RND 20	20 * 20	2276	**2268**
RND 25	25 * 25	**2920**	2953

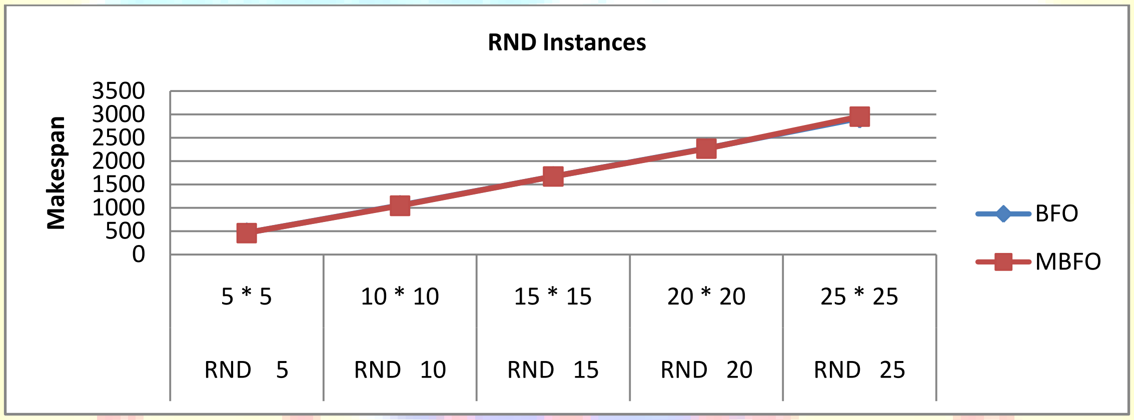


Figure III: Graphical representation of results for RND variable size Instances

C. Comparison Results for RND 6x6 Instances

BFO algorithm is compared with proposed MBFO algorithm of RND instance sizes 6x6 are shown in Table IV. The figure IV shows the graphical representation of Table IV.

Table IV: Comparison Results of BFO & MBFO for RND 6x6 Instances

INSTANCE	SIZE	BFO	MBFO
RND 1	6 * 6	**528**	531
RND 2	6 * 6	505	**502**
RND 3	6 * 6	542	**539**
RND 4	6 * 6	**531**	554

RND 5	6 * 6	570	**564**

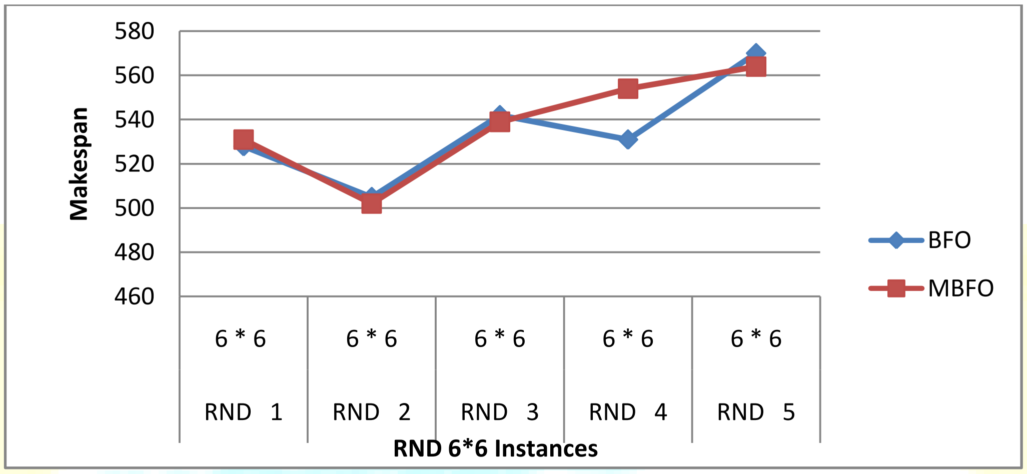


Figure IV: Graphical representation of results for RND 6x6 Instances

D. Comparison Results for RND 6x6 Instances

BFO algorithm is compared with proposed MBFO algorithm of RND instance sizes 7x7 are shown in Table V. The figure V shows the graphical representation of Table V.

Table V: Comparison Results of BFO & MBFO for RND 7x7 Instances

INSTANCE	SIZE	BFO	MBFO
RND 1	7 * 7	**672**	673
RND 2	7 * 7	613	**608**
RND 3	7 * 7	685	**681**
RND 4	7 * 7	**613**	625
RND 5	7 * 7	652	**642**

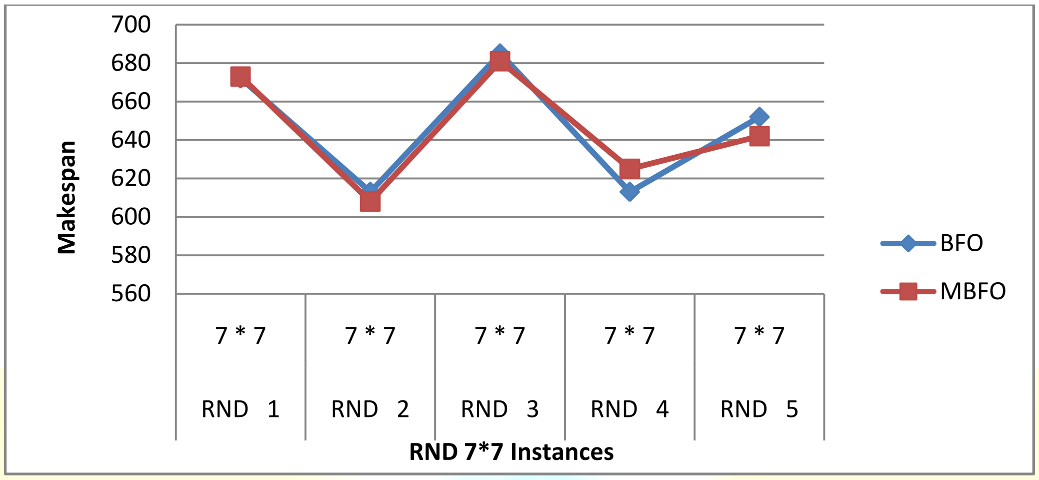


Figure V: Graphical representation of results for RND 7*7 Instances

V. CONCLUSIONS

Flow Shop Scheduling Problems are mainly concerned with completion time related objectives, however, in modern manufacturing and operations management, on time delivery is a significant factor as for the reason of upward stress of competition on the markets. The proposed MBFO algorithm gave the best Makespan for most of Carlier Instances and RND Instance when compared with BFO algorithm. The implementation of the MBFO for large size instances can be done by increasing the number of iterations to achieve optimal solutions. The proposed MBFO for FSSP can be improved to achieve optimal solution by including the swarming technique and also by modifying constant values used in the algorithms. As a future work, Flexible Flow Shop Scheduling problems can also be solved using proposed MBFO algorithm.

REFERENCES

1. AndreasFink, StefanVoß., *Solving the Continuous Flow-Shop Scheduling Problem by Metaheuristics*, 2001.
2. Arijit Biswas, Sambarta Dasgupta, Swagatam Das, and Ajith Abraham., *Synergy of PSO and Bacterial Foraging Optimization – A Comparative Study on Numerical Benchmarks*, 2007.
3. Arnaud Malapert, Hadrien Cambazard, Christelle Guéret, Narendra Jussien and André Langevin., Louis-Martin Rousseau, *An Optimal Constraint Programming Approach to the Open-Shop Problem*.
4. Ashwani Dhingra and Pankaj Chandna., *Hybrid Genetic Algorithm for Multicriteria Scheduling with Sequence Dependent Set up Time*.
5. Ashwani Kumar Dhingra., *Multi-Objective Flow Shop Scheduling using Metaheuristics*.
6. Ching-Fang Liaw., *A hybrid genetic algorithm for the open shop scheduling problem,* 1999.
7. Christelle Guéret, Christian Prins, Marc Sevaux., *Applications of optimization with Xpress-MP*,2000.
8. Chunguo Wu, Na Zhang, Jingqing Jiang, Jinhui Yang, and Yanchun Liang., *Improved Bacterial Foraging Algorithms and their Applications to Job Shop Scheduling Problems*, 2007.
9. David Applegate, William Cook., *A Computational Study of the Job-Shop Scheduling Problem*, 1991.
10. E. Taillard., *BenchMarks For Basic Scheduling Problems*, 1989.
11. H. Van Dyke Parunak., *"Go to the Ant" Engineering Principles from Natural Multi-Agent Systems*, 1997.
12. Hela Boukef, Mohamed Benrejeb and Pierre Borne., *A Proposed Genetic Algorithm Coding for Flow-Shop Scheduling Problems*, 2006-2007.
13. Ilkyeong. Moon, Jieom. Lee., *Genetic Algorithm Application to the Job Shop scheduling Problem with Alternative Routings*, 2000
14. James Montgomery, cardc Fayad and Sarja Petrovic., *Solution Representation for Job Shop Scheduling Problems in Ant Colony Optimization.*
15. Jing Dang, Anthony Brabazon, Michael O'Neill, and David Edition., *Option Model Calibration using a Bacterial Foraging Optimization Algorithm*, 2008.

16. Jun Zhang, Xiaomin Hu, X.Tan, J.H Zhong and Q. Huang., *Implementation of an Ant Colony Optimization Technique for Job Shop Scheduling Problem*, 2006.
17. Katie Kinzler., *Mathematical Modeling of Ant Pheromones: Determination of Optimum pheromone Evaporation Rate and Simulation of Pheromone Tracking Abilities*, 2008.
18. Kevin Passino., *E-coli Bacterial Foraging for Optimization.*
19. Mahanim Omar, Adam Baharum, Yahya Abu Hasan., *A Job-Shop Scheduling Problem (JSSP) using Genetic Algorithm*, 2006.
20. Marcelo Seido Nagano., *A Constructive Genetic Algorithm fo r Permutation Flowshop Scheduling*.
21. O. Seraj and R. Tavakkoli-Moghaddam., *A Tabu Search Method For A New Bi-Objective Open Shop Scheduling Problem By A Fuzzy Multi-Objective Decision Making Approach*.
22. Orhan ENGİ N, Alper DÖYEN ., *A New Approach To Solv E Flowshop Scheduling Problems By Artificial Immune Systems*,2007.
23. R.Murugesan, S.ThamaraiSelvi., P.Alphonse Rajendran and V.S.SampathKumar., *Identication of a Rank Minimal Optimal Sequence for Open Shop Scheduling Problems*, 2003.
24. Rutger Claes and Tom Holvoet., *Cooperative Ant Colony Optimization in Traffic Route Calculations*.
25. S. Subramanian and S. Padma., *Bacterial Foraging Algorithm Based Multiobjective Optimal Design of single phase Transformer*, 2011.
26. Samia kouki, Mohamed Jemni and Talel Ladhari., *Solving the Permutation Flow Shop Problem with Makespan Criterion using Grids*, 2011.
27. Seda HEZER, Yakup KARA., *Solving Vehicle Routing Problem with Simultaneous Delivery and Pick-up using Bacterial Foraging Optimization Algorithm*.
28. Teofilo Gonzalez and Sartaj Sahni., *Open Shop Scheduling to Minimize Finish Time*, 1976.
29. Vittorio Maniezzo, Luca Maria Gambardella and Fabio de Luigi., *Ant Colony Optimization*.
30. http://people.brunel.ac.uk/~mastjjb/jeb/info.html

Author's Profile:

Ms. P. Divya received her B. Sc Degree in Computer Science, Masters Degree (MCA) in Computer Applications in 2009 and 2012 respectively from Bharathiar University, Coimbatore, Tamil Nadu, India. Her area of interest is Bio-inspired computing.

Mrs. T. Amudha received her B.Sc Degree in Physics, Masters Degree (MCA) in Computer Applications and M.Phil in Computer Science in 1995, 1999 and 2003 respectively, from Bharathidasan University, India. She has qualified UGC-NET for Lectureship in 2003 and is currently pursuing her doctoral research at Bharathiar University in the area of Agent Systems. She is currently working as Asst. Professor in the Department of Computer Applications, Bharathiar University, Coimbatore and has 13 years of academic experience. She has more than 20 research publications for her credit in International/ National Journals & Conferences. Her area of interest includes Software Agents, Bio-inspired computing and Grid computing.

Mr. S. Narendhar received his B. Sc Degree in Computer Technology from Anna University, Coimbatore, India in the year 2009 and Masters Degree (MCA) in Computer Applications from Bharathiar University, Coimbatore, India in the year 2012. His area of interest includes Agent based computing and Bio-inspired computing. He has attended National Conferences.