An approach of a Lampyridae family (firefly) algorithm for optimisation of Bagchi's job shop scheduling problems

Kuppuswamy Chockalingam Udaiyakumar*

Department of Mechanical Engineering,
Sathyabama University,
Chennai – 119, India
Email: kcudaiyakumar1967@gmail.com
*Corresponding author

M. Chandrasekaran

Department of Mechanical Engineering,
Vels University,
Chennai – 117, India
Email: chandrasekar2007@gmail.com

Abstract: Today, every organisation finds it a great challenge to fulfil the needs of its customers. In order to gratify their requirements of its clients. It is imperative for the organisations to integrate product design and development. In this process, scheduling plays a vital role. Scheduling problems can be solved using traditional methods in general and also involves huge computational difficulty and time consuming. From the literature review, it is inferred that by using traditional methods involves a huge difficulty in solving high complex problems and metaheuristic algorithms were proved to be most efficient algorithms to solve various job shop scheduling problems. The objective of this paper is to apply a recently developed metaheuristic algorithm also known as fire-fly algorithm to find optimal makespan and mean flow time of different size problems using to Bagchi job shop scheduling problems called JSP1 and JSP2 and also to prove that a proposed algorithm serves a good problem solving technique for JSSP with multi criteria.

Keywords: job shop scheduling problem; JSSP; fire-fly; makespan; mean flow time; benchmark.

Reference to this paper should be made as follows: Udaiyakumar, K.C. and Chandrasekaran, M. (2015) 'An approach of a Lampyridae family (firefly) algorithm for optimisation of Bagchi's job shop scheduling problems', *Int. J. Enterprise Network Management*, Vol. 6, No. 4, pp.324–339.

Biographical notes: Kuppuswamy Chockalingam Udaiyakumar is an Assistant Professor (selection grade) in the Department of Mechanical Engineering for the last five years in one of the prestigious university SRM University in Chennai. He has been holding the responsibility of Head of the department since 2010. Currently, he is pursuing his research work registering in Sathyabama University. He has attended more than five international conferences and published three papers in international journals and also guiding PG students project and helped to develop most of the laboratories in modernised way.

M. Chandrasekaran is the Director in the Department of Mechanical Engineering for the past three years in Vels University. He is guiding more than 20 research scholar in various leading universities in Tamil Nadu. He has published more than 30 papers in international journals. He has organised two international conferences at Vels University.

This paper is a revised and expanded version of a paper entitled 'Solving of multiobjective job shop scheduling problems with firefly algorithm' presented at ICMOC 2014, Noorul Islam Universtiy, Kumaran Koil, Nagercoil, Tamil Nadu, India, 10–11 April 2014.

1 Introduction

In the modern world a rapid changes are not only happening in manufacturing industries and happenings with other industries as well. The manufacturing industry contribute significantly for economic growth and development of a country. In manufacturing industries scheduling is considered to be a major task for shop floor productivity improvement. A schedule is an assignment of operations to time slots on the relevant machines. There are three kinds of scheduling namely

1 single machine scheduling
2 floor shop scheduling and
3 job shop scheduling.

Single machine scheduling problem (Pannerselvam, 2010) consists of n jobs with the same single operation on each of the jobs, while the flow shop scheduling problem consists of n jobs with m operations on each of the jobs. In this problem all the jobs will have the same process sequences. The job shop scheduling problems (JSSPs) contains n jobs with m operations on each of the jobs; but in this case the process sequence of the jobs will be different from each other.

In Flow shop scheduling, there are n jobs; each requires processing on m different machines. The order in which the machines are required to process a job is called process sequence. The process sequences of all the jobs are the same. But the processing time for various jobs on a machine may differ. If an operation is absent in a job, then the processing time of the operation of that job is assumed as zero.

In job shop problem, it is assumed that each job has m different operations. In some cases if the jobs are having less than m operations, required number of missing operations with zero process time is assumed as dummy operations. By this assumption, the condition of equal number of operations for all the jobs is ensured. In JSSP process sequences of the jobs are not the same. Hence the flow of each job in job shop scheduling is not unidirectional.

1.1 Job shop scheduling problem

Scheduling is the allocation of resources overtime to perform a collection of tasks. The JSSP consists of set of m machines$\{M_{1,}\ M_{2\ldots}\ M_n\}$, and a collection of n jobs

$\{J_1, J_2, \ldots\ldots\ldots J_n\}$ to be scheduled, where each job must pass through each machine only once. Each job has its own processing order and this may bear no relation to the processing order of any other job. The JSSPs are NP-hard problem, which are complex in nature (Chandrasekaran et al., 2007).

The following assumptions are made while solving job shop problem.

1 each job requires m machines to complete the required process

2 the operations can be processed in any order

3 there is no parallel processing.

1.2 Single objective job shop model

Single objective job shop model consists of minimisation of makespan as an objective function which is given in equation (1), Sequence and resource constraints are shown in equations (2), (3) and (4), respectively. The makespan means the completion time of all the jobs which is considered for scheduling. The Job shop i(i = 1, 2, 3, ……. n) requires processing by machine k(k = 1, 2,…., m) exactly once in its operation sequence. Let p_{ik} is the processing time of job i on machine k, X_{ik} is the starting time of job I on machine k, q_{ijk} is the indicator which takes on a value of 1 if operation j of job i requires machine k, and zero otherwise. Y_{ihk} is the variable which takes on a value of 1 if job i precedes job h on machine k, and zero otherwise.

The objective function is minimisation of makespan

$$\text{Minimise } Z = \sum_{k=1}^{m} q_{imk}\left(x_{ik} + p_{ik}\right) \tag{1}$$

Subject to

a Sequence constraint

$$\sum_{k=1}^{m} q_{imk}\left(x_{ik} + p_{ik}\right) \le \sum_{k=1}^{m} q_{i,j+1,k} X_{ik} \, (i = 1, \ldots n; j = 1, \ldots m - 1) \tag{2}$$

i.e., for a given job i, the (j + 1)st operation may not start before the j^{th} operation is completed.

b Resource constraint

$$X_{hk} - X_{ik} \ge p_{ik} - \left(H + p_{ik}\right)\left(1 - Y_{ihk}\right) \tag{3}$$

$$X_{ik} - X_{hk} \ge p_{hk} - \left(H + p_{hk}\right) Y_{ihk} \tag{4}$$

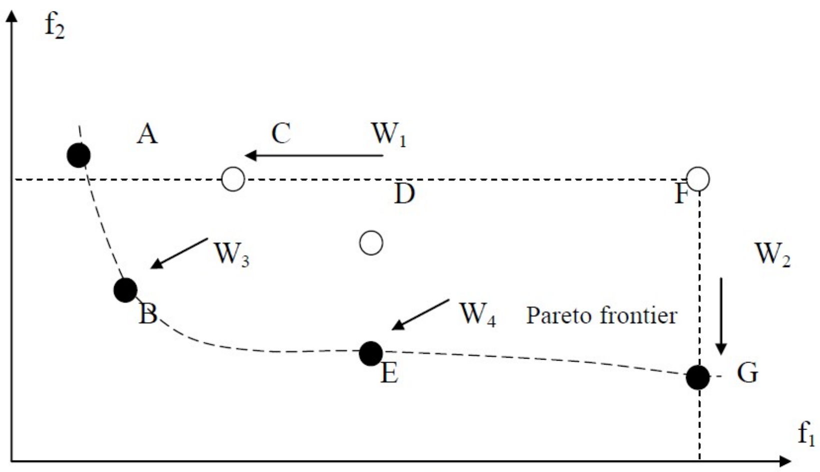
where (i = 1,….n; h = 1,….n; k = 1,……….m) where H is a very large positive integer.

1.3 Multi objective job shop model

1.3.1 Multi-objective approach

The scheduling problems are generally multi objective in nature. Multi objective scheduling problems are complex when compared to single objective categories. Multi objective optimisation differs from single objective optimisation in several ways (Deb 1999). In such cases many objectives are considered all together when schedule is generated. Thus the goal is to generate a feasible schedule that minimises many objectives. This schedule is called a Pareto optimal solution. For two or more contradictory objectives, each objective corresponds to different optimal solutions but none of these trades off solutions is optimal with respect to all objectives. Hence multi objective optimisation does not try to find out one optimal solution but optimises all trade off solutions. The multi objective optimisation deals with two goals, the first goal is to explore for a set of solutions as close as possible to Pareto-optimal front and the second goal is to find a set of solutions as diverse as possible. A single feasible schedule that optimises many objectives may not exist. It means that individual optimal solutions of each objective are usually different. Under such circumstances, a schedule with weighted combination of many scheduling objectives is considered. It is possible that weights of objectives are known before scheduling. This approach permits computing of a unique strict Pareto optimal solution. The set of Pareto solutions is called the Pareto front. Therefore solving a multiobjective scheduling problem is a Pareto optimisation problem. The Pareto optimality approach with weightage is shown in Figure 1.

Figure 1 Pareto optimality approach



1.3.2 The second objective function is mean flow time

1.3.2.1 Mean flow time

Let w_i denote the weight assigned to the i^{th} job in a batch of n jobs given and F_i denotes flow time of i^{th} job, the weighted mean flow time is defined as,

$$\frac{\frac{1}{n} * \sum_{i=1}^{n} w_i f_i}{\sum_{i=1}^{n} w_i} \tag{5}$$

The usual problem considered is minimising the (weighted) mean flow time, which is equivalent to minimising the total flow time.

Two jobs to be performed by three machines: (2 × 3) JSSP is illustrated in Table 1. In this problem, each job requires three operations to be processed on a pre-defined machine sequence. The first job (J_1) needs to be initially operated on machine M_1 for 10 time units and then sequentially processed on M_2 and M_3 for 9 and 8 time units, respectively. Likewise, the second job (J_2) has to be initially performed on M_3, M_1 and M_2 for 9, 8, 7 time units respectively. The design task for solving JSSP is to search for the best schedule(s) for operating all pre-defined jobs in order to optimise either single or multiple scheduling objectives. An example of two jobs three machines scheduling problem with processing times.

Table 1 Two job three machines scheduling problem with processing time

Job	*Operation*	*Time*	*Machine (M_k)*		
	(O_{jk})	*(t_{jk})*	*M_1*	*M_2*	*M_3*
J_1	0_{11}	10	10	-	-
	0_{12}	9	-	9	-
	0_{13}	8	-	-	8
J_2	0_{23}	9	-	-	9
	0_{21}	8	8	-	-
	0_{22}	7	-	7	-

2 Literature survey

Various optimisation approaches have been widely applied to solve the JSSP. Conventional methods based on either mathematical methods or full numerical search (for example, branch and bound and Lagrangian relaxation can guarantee the optimal solution. They have been successfully used to solve the JSSP. However, these methods highly consume computational time and resources becomes even for solving moderately-large problem size and therefore impractical if the computational limitation exists. Later, a larger size JSSP have been solved by an approximation optimisation methods or meta-heuristics such as tabu search and simulated annealing. The summarised literature survey on various previous research papers as shown in Table 2. From the review of literature by the application of this algorithm many researchers experimentally proved in their work this algorithm outperformed (Hashmi et al., 2013) other meta heuristics in many fields. In the engineering design approach, Gandomi et al. (2011) and Azad et al. (2011) proved that firefly algorithm can successfully solve highly nonlinear, multimodal design problems. Sayadi et al. (2010) developed a discrete firefly meta-heuristic with local search for makespan minimisation in permutation flow shop scheduling problems. In the recent past, Yang and He (2013) concluded in their work that this algorithm is better than the optimal intermittent search strategy.

Table 2 Summary of various researchers approach on JSSP

Category	*Approaches*	*Authors*	*Work approach*	*Outcome*
Heuristic rules	Dispatching rules	Lawrence (1984)	Presented ten new dispatching rules for scheduling in a job shop with randomised combination of rules	More suitable as an Initial solution technique
		Jeremiah et al. (1964)	Proposed a dispatching rule that combines multiple dispatching criteria into a single rule	Used for makespan performance measure
Mathematical programming	Branch and Bound	Artigues and Feillet (2007)	Proposed a new method based on iterative approach via branch and bound decisional versions	This method allows to solve previously unsolved benchmark
		Artigues et al. (2007)	Proposed a insertion heuristic and generalises resource constraint propagation mechanism	Results prove the constraint propagation generalisation when time lags are considered
	Lagrangian relaxation	Baptiste et al. (2008)	Compared two Lagrangian relaxation approach for the job shop	These results show that these problems are greatly intractable
		Chen and Luh (2003)	Developed a new Lagrangian relaxation approach for JSSP	New approach is efficient for large problems with long time horizons
Neighbourhood search methods	Tabu search	Ponnabalam et al. (2000)	TS method proved and reported quite favourably with benchmark results	TS implemented most effectively. Most of benchmark result found in this paper
	Simulated annealing (SA)	Mohanasundaram et al. (2005)	Proposed Pareto archived simulated annealing	The proposed approach applied to solve multi objective in JSP

Table 2 Summary of various researchers approach on JSSP (continued)

Category	*Approaches*	*Authors*	*Work approach*	*Outcome*
Artificial intelligence	Artificial neural network	Yang et al. (2010)	Presented an improved constraint satisfaction adaptive neural network (NN) for JSSP	Outperform the original adaptive NN in term of computational time and quality of schedules
		Khaw et al. (1991)	Proposed a new back error propagation approach implemented for JSSP	It helps to receive variable data such as job orders capacity availability and set up times while outputting machine orders
	Fuzzy logic	Krucky (1994)	In this approach to combine dispatching rules	Proposed algorithm addressed minimising the set up times of medium to high product mix production lines
Nature inspired algorithm	Ant colony algorithm	Denebourg et al. (1983)	ACO-based stochastic optimisation technique was developed	Ants behaviour helps to find the shortest path, is a simple cooperating agent for search performance, allowing poor local minima to be transcended
	Sheep flocks heredity model algorithm	Chandrasekaran et al. (2007)	In this proposed algorithm used multi stage genetic operation, it means that sub chromosome level genetic operation and chromosome (global) level genetic operation	Approached for single objective (minimisation of makespan) and this algorithm also extended for multi objective problem (makespan and tardiness)
	Firefly algorithm	Khadwilard et al. (2012)	An approach to set up optimised parametric settings by ANOVA	Applied to determine makespan

3 Firefly algorithm

3.1 Firefly algorithm

Firefly algorithm was developed by Yang (2009), Fireflies, belongs to the family of Lampyridae, are tiny winged beetles having capability of producing light. Firefly algorithm idealises some of the characteristics of the firefly behaviour. They follow three rules:

a all the fireflies are unisex so that one firefly will be attracted to other fireflies regardless of their sex

b each firefly is attracted only to the fireflies, that are brighter than itself; Strength of the attractiveness is proportional to the firefly's brightness, which attenuates over the distance; the brightest firefly moves randomly

c brightness of every firefly determines it is quality of solution; in most of the cases, it can be proportional to the *objective function.*

Using the above three rules, a pseudo-code of the firefly algorithm may look as follows:

Algorithm 1 Basic firefly algorithm pseudo-code

```
Input: f(x), x = (x_1, x_2 .........., x_n); //
Objective function
    n, I_0, ϒ, α; // User-defined constants
Output: x_mean, ft_mean // position of minimum in objective function
for i ← 1 to n do
  | x_i ← Initial_Solution ( );
end
While termination requirements are not met do
  | min ← arg min (f(x_i));
  |   | i ε {1, ........., n}
  | for i ← 1 to n do
  |   | for j ← 1 to n do
  |   |   | if f(x_i) < f (x_j) then
  |   |   |   | d_i,j ← Distance(x_i, x_j); //move x_i towards x_j
  |   |   |   | β ← Attractiveness (I0, ϒ,di,j);
  |   |   |   | X_i ← (1 – β) X_i + β X_j + α (Random ( ) – 1/2);
  |   |   |   | // movemen
  |   | end
  | end
x_mean, ft_mean ← x_min + α (Random ( ) – 1/2);// best briefly moves randomly
end
```

In the above algorithm, n is the number of the fireflies, I_0 is the light intensity at the source, γ is the absorption coefficient, β is the attractiveness and α is the size of the random step. X_{min} is minimum makespan and ft_{mean} is mean flow time respectively.

3.2 Application of firefly algorithm for job shop scheduling

3.2.1 Introduction

The objective of this paper is to apply a newly developed metaheuristic algorithm also known as firefly algorithm to find optimal makespan and mean flow time of different size problems using to Bagchi JSSPs called JSP1 and JSP2 and also to prove that a proposed algorithm serves a good problem solving technique for JSSP with multi criteria.

3.2.2 Firefly evaluation

The next stage is to measure the flashing light intensity of the firefly, which depends on the current problem considered. In this work, the evaluation on the correctness of the schedules is measured by the makespan, which can be calculated using equation (6), where C_k is completed time of job k.

$$\text{Minimises } C_{max} = \max\left(C_1,\ C_2, C_3, \ldots\ldots.. C_k\right) \tag{6}$$

3.2.3 Distance

The distance between any two fireflies i and j at X_i and X_j, respectively, can be defined as Cartesian distance (r_{ij}) using equation (7), where $X_{i,k}$ is the component of the spatial coordinate x_i of the i^{th} firefly and d is the number of dimension.

$$r_{ij} = \left\| x_i - x_j \right\| = \sqrt{\sum_{k=1}^{d}\left(\left(x_{i,k} - x_{j,k}\right)^2\right)} \tag{7}$$

3.2.4 Attractiveness

The calculation of attractiveness function of a firefly are shown in equation (8), where r is the distance between any two fireflies,β_0 is the initial attractiveness r = 0, and γ is an absorption coefficient which controls the decrease of the light intensity explained by (Yang, 2009),

$$\left(\beta_{(r)}\right) = \beta_0 * \exp\left(-\gamma^{\gamma,m}\right), \text{ with } m \geq 1 \tag{8}$$

These recently developed algorithms have been applied by few researchers for solving optimisation problems. In this work, the settings of FFA parameter such as number of fireflies (n), number of generations/iterations (G), the light absorption coefficient (γ), randomisation parameter (α) and attractiveness value (β_0) have to be chosen in an ad hoc fashion. Generally the combination factor (nG) determines the amount of search in the solution space conducted by this algorithm. This factor is directly related with the size of the problem considered. In this research, the acceptable computational limitations are practically implemented; therefore the combination factor was fixed at 1,000 in order to accommodate computational search within the time limit. The light absorption coefficient (γ) was varied from 0 to 10, the randomised parameter was usually set between 0 to 1 and the attractiveness function was also chosen between 0 to 1.

Table 3 Due dates and processing times for JSP problem #1

	(m/c, t)	*(m/c, t)*	*(m/c, t)*	*(m/c, t)*	*(m/c, t)*	*Due date*
Job 1	1,13	5, 16	4, 19	2, 7	3, 14	37
2	4, 19	5, 7	2, 13	1, 17	3, 19	74
3	3, 19	2, 18	5, 16	4, 18	1, 19	111
4	1,14	4, 15	5, 10	2, 13	3,17	148
5	1,8	2, 8	5, 19	4, 7	3,9	185
6	3, 16	2, 15	5,20	4, 18	1, 10	222
7	2, 14	4, 17	3, 18	1,5	5, 20	259
8	1,8	2,6	4, 9	5, 20	3, 7	296
9	5, 16	4, 13	3, 9	2, 16	1, 12	333
10	2, 12	1, 19	3, 9	5, 6	4, 7	370

4 Experimental result and analysis

In order to solve the optimisation problem this research work is applied to Bagchi's(1999) two JSP problems JSP1 and JSP2 as shown in Tables 4 and 5, to accomplish it in MATLAB under Windows XP operating system. The following parameter is used in solving both the JSP problem by sensitivity analysis as shown in Table 6, are $\alpha = 0.005$, $\beta_0 = 1.0$, $\gamma = 1.0$, number of fireflies (n) is 10 and maximum generation (G) of fireflies is 100 hence total no of functional evolution (nG) is 1,000. The results of computational experiments for two Bagchi problems (JSP1 and JSP2) are shown in Tabled 6 and 7 and Figures 2 to 5.

The results are also compared with other algorithm results by previous researcher as shown in Table 6 (JSP1) and Table 7 (JSP2).

From Table 6 and Figures 2 and 3, it is observed that for the JSP 1 out of ten solutions for the first objective minimum makespan six solutions are exactly matched with best known solutions (BKS) and remaining four solutions are near to BKS. Where as in another objective mean flow time out of ten solutions four solutions exactly matched and remaining were near to optimal. Similarly from Table 7 and Figures 4 and 5, for the JSP2 it is found that for the both objectives out of ten solutions eight solutions were exactly matching with BKS (Deb, 1999) and remaining were lesser to BKS. Owing to its size of the problems and resources available.

Table 4 Due dates and processing times for JSP problem #2

	(m/c, t)	*(m/c, t)*	*(m/c, t)*	*(m/c, t)*	*(m/c, t)*	*(m/c, t)*	*(m/c, t)*	*(m/c, t)*	*(m/c, t)*	*(m/c, t)*	*Due date*
Job 1	2, 13	9, 16	1, 17	3, 7	6, 14	7, 19	4, 13	8, 9	10, 12	5, 20	34
2	7, 12	5, 19	10, 5	9, 6	8, 15	4, 15	6, 12	2, 8	1, 6	3, 12	68
3	8, 8	7, 8	6, 10	9, 20	4, 12	1, 10	3, 15	5, 15	10, 5	2, 17	102
4	3, 8	10, 20	4, 14	2, 19	8, 17	6, 18	5, 9	9, 20	7, 6	1, 8	136
5	5, 20	1, 13	3, 12	8, 7	9, 9	7, 10	2, 8	10, 16	4, 6	6, 16	170
6	8, 6	3, 14	10, 19	4, 14	9, 20	2, 12	1, 16	8, 9	7, 16	5, 6	204
7	2, 16	3, 6	8, 7	10, 17	7, 13	1, 17	5, 17	6, 17	4, 14	9, 15	238
8	4, 66	1, 3	3, 14	1, 9	10, 13	5, 13	7, 19	2, 15	8, 18	9, 16	272
9	7, 8	2, 13	3, 13	9, 7	10, 9	5, 10	6, 16	1, 11	4, 16	8, 18	306
10	3, 13	4, 18	1, 19	5, 6	10, 5	9, 19	2, 19	6, 6	8, 9	7, 18	340

Table 5 Comparison of FFA parameters setting used in previous researchers

Authors	*Problems*	*FFA parameters*			
		nG	*Γ*	*A*	*β0*
Lukasik and Zak (2009)	Continuous equation	40 * 250	1.0	0.01	1.0
Khadwilard et al. (2012)	Job shop scheduling	100 * 25	0.1	0.5	1.0
Our proposal	Job shop scheduling	10 * 100	0.0001	0.05	1.0

Table 6 Comparison of JSP 1 Pareto solutions of FFA with GA

Solutions	*Makespan*		*Mean flow time*	
	GA	*FFA*	*GA*	*FFA*
1	159	159	120	124.3
2	167	167	122	122.4
3	182	182	132	135.7
4	156	158	126	128.4
5	169	169	132	134.5
6	159	160	126	127.3
7	160	160	124	124.3
8	165	165	126	128.8
9	158	158	126	126
10	162	165	130	130.5

Table 7 Comparison of JSP2 Pareto solutions of FFA with GA

Solutions	*Makespan*		*Mean flow time*	
	GA	*FFA*	*GA*	*FFA*
1	196	195	174.7	162.1
2	199	199	174.6	174.6
3	204	204	174.8	174.8
4	207	207	176.6	176.6
5	209	209	173.4	173.4
6	212	212	174.5	174.5
7	228	215	179.4	164.7
8	230	230	179.4	179.4
9	238	238	188.1	188.1
10	254	254	186.7	186.7

Figure 2 Comparison of makespan with GA and FFA (see online version for colours)

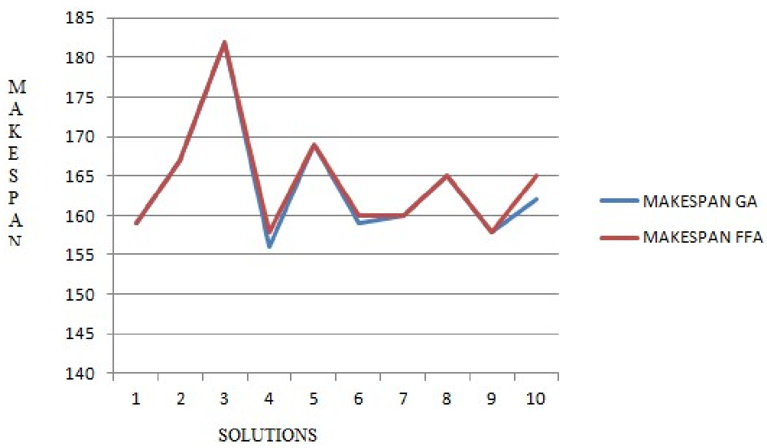


Figure 3 Comparison of meanflow with GA and FFA (see online version for colours)

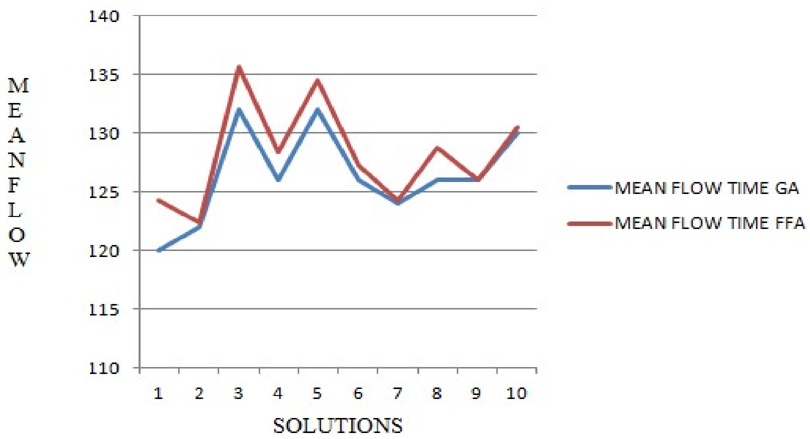


Figure 4 Comparison of makespan with GA and FFA (see online version for colours)

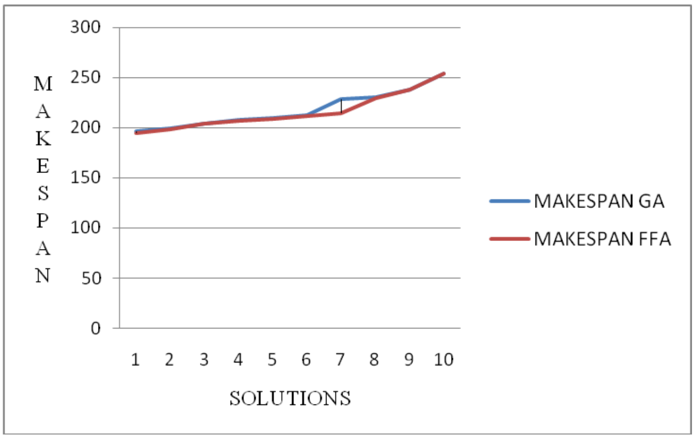
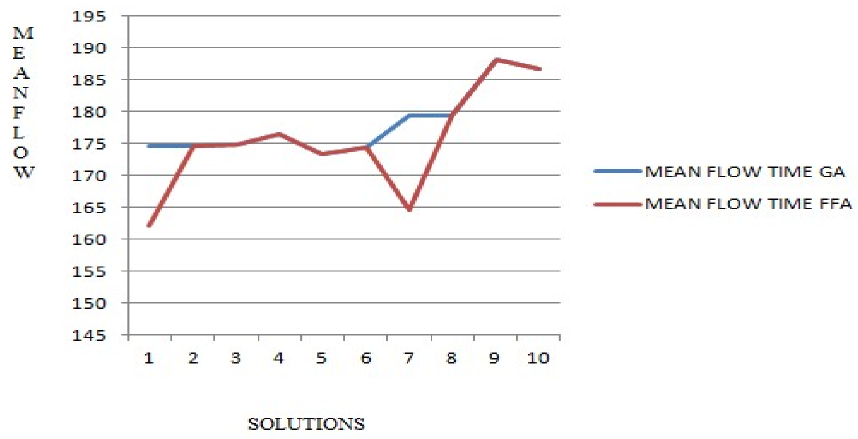


Figure 5 Comparison of meanflow with GA and FFA (see online version for colours)



5 Conclusions

Thus, recently developed metaheuristics have gained popularity owning to the ability with nonlinear global optimisations. In this paper it is demonstrated that the recently developed firefly algorithm is a powerful technique used to solve the problems of job shop scheduling optimisation. It is one of the simplest method and easy to apply on any Non-polynomial hard problem. The minimisation of makespan (C_{max}) and mean flow time using two Bagchi JSP problem can be interpreted using the algorithm. This algorithm was applied to find the Pareto optimal solutions of two Bagchi's JSP1 and JSP2 problems. The parameters of FFA such as absorption coefficient, population of the firefly and number of iterations depends upon the optimised problem. In the result of both Bagchi's problems, it is observed that the first case (JSP1) almost six solutions out of ten solutions (makespan) matched with best known solutions where as in second case (JSP2) both the objectives related to solutions were almost matched but few were found less than best known solution (GA) it is quite clear that this algorithm is one of the best heuristics approaches to solve multi objective criteria in JSSP. This work paves way for further research to optimised above problem further and any bench marking problem of JSSP by varying the above controlling parameters. There is no doubt that the firefly can be applied in solving more challenging problems in future.

References

Artigues, C. and Feillet, D. (2007) 'A branch and bound method for the job-shop problem with sequence dependent setup times', *Annals of Operations Research*, Vol. 159, No. 1, pp.135–159.

Artigues, C., Huguetand, M-J. and Lopez, P. (2007) 'Generalized disjunctive constraint propagation for solving the job shop problem with time lags', *Engineering Applications of Artificial Intelligence*, Vol. 24, No. 2, pp.220–231.

Azad, S.K and Saeid, S.K. (2011) 'Optimum design of struces using an improved firefly algorithm', *International Journal of Optimization in Civil Engineering*, Vol. 1 (satisfaction 2), No. 2, pp.327–340.

Bagchi, T.P. (1999) *Multi Objective Scheduling by Genetic Algorithms*, Kluwer Academic Publisher, Norwell, MA.

Baptiste, P., Flamini, M. and Sourd, F. (2008) 'Lagrangian bounds for just-in-time job-shop scheduling', *Computers & Operations Research*, Vol. 35, No. 3, pp.906–915.

Chandrasekaran, M., Ashokan, P., Kumanan, S. and Umamaheswari, S. (2007) 'Multi objective optimization of job shop scheduling using sheep flocks heredity model algorithm', *International Journal of Manufacturing Science and Technology*, USA, Vol. 9, No. 2, pp.47–54.

Chen, H.X. and Luh, P.B. (2003) 'An alternative framework to Lagrangian relaxation approach for job shop scheduling', *European Journal of Operational Research*, Vol. 149, No. 3, pp.499–512.

Deb, K. (1999) *Multi Objective Optimization using Evolutionary Algorithms*, 2nd ed., Wiley, Chichester, UK.

Denebourg, J.L., Pasteels, J.M. and Verhaeghe, J.C (1983) 'Probabilistic behaviour in ants: a strategy of errors?', *Journal of Theoretical Biology*, Vol. 105, No. 2, pp.259–271.

Gandomi, A.H., Yang, X.S. and Alavi, A.H. (2011) 'Cuckoo search algorithm: a meta heuristic approach to solve structural optimization problems', *Engineering with Computers*, Vol. 27, article DOI 10.1007/s00366-011-0241-y.

Jeremiah, B., Lalchandani, A. and Schrage, L. (1964) *Heuristic Rules Toward Optimal Scheduling*, Research report, Department of Industrial Engineering, Cornell University.

Khadwilard, A., Chansombat, S. et al. (2012) 'Application of firefly algorithm and its parameter setting for job shop scheduling', *The Journal of Industrial Technology*, Vol. 8, No. 1, pp.49–58.

Khaw, J., Siong, L. B., Lim, L., Yong, D., Jui, S.K. and Fang, L.C. (1991) 'Shop floor scheduling using a three-dimensional neural network model', *International Conference on Computing Integrated Manufacturing*, September 30–October 4, Singapore, pp.563–566.

Krucky, J (1994) 'Fuzzy family setup assignment and machine balanching', *Hewlett-Packard Journal*, June, Vol. 45, No. 3, pp.51–64.

Lawrence, S. (1984) *Supplement to Resource Constrained Project Scheduling: An Experimental Investigation of Heuristic Scheduling Techniques*, Graduate School of Industrial Administration, Carnegic-Mellon University, Pittsburgh, USA.

Lin, T-L., Horng, S-J., Kao, T-W., Chen, Y-H., Run, R-S., Chen, R-J., Lai, J-L. and Kuo, I.H. (2010) 'An efficient job-shop scheduling algorithm based on particle swarm optimization', *Expert Systems with Applications*, Vol. 37, pp. 2629–2636.

Lukasik, S. and Zak, S. (2009) 'Firefly algorithm for continuous constrained optimization tasks', *Lecture Notes in Computer Science*, Vol. 5796, pp 97–106.

Mohanasundaram, K.M. et al. (2005) 'Pareto archived simulated annealing for job shop scheduling with multiple objectives', *International Journal of Advanced Manufacturing Technology*, Vol. 167, No. 1, pp.77–95.

Ono, I., Yamunara, M. and Kobayashi, S. (1996) 'A genetic algorithm for job shop scheduling problems using job based order crossover', in *Proceedings of ICEC*, pp.547–552.

Pannerselvam, R. (2010) *Production and Operation Management*, 2nd ed., pp.312–349, PHI Learning Private Limited, New Delhi.

Ponnabalam, S.G., Aravindhan, P. and Rajesh, S.V. (2000) 'A tabu search algorithm for job shop scheduling', *International Journal of Advanced Manufacturing Technology*, Vol. 16, pp.765–771.

Ripon, K., Tsang, C-H. and Kwong, S. (2007) 'An evolutionary approach for solving the multi-objective job-shop scheduling problem', *Studies in Computational Intelligence*, Vol. 49, pp.165–195.

Sayadi, M.K, Ramezian, R. and Ghaffari-nasab, N. (2010) 'A discrete firefly meta-heuristic with local search for makespan minimization in permutation flow shop scheduling problems', *International Journal of Industrial Engineering Computations*, Vol. 1, No. 1, pp.1–10.

Yang, X.S. (2009) 'Firefly algorithms for multimodal optimization, stochastic algorithms: foundations and applications', *SAGA, Lecture Notes in Computer Sciences*, Vol. 5792, pp.169–178.

Yang, X.S. and He, X. (2013) 'Firefly algorithm; recent advances and applications', *International Journal Swarm Intelligence*, Vol. 1, No. 1, pp.36–50.

Yang, X.S., Wang, D., Chai, T. and Kendall, G. (2010) 'An improved constraint satisfaction adaptive neural network for job-shop scheduling', *Journal of Scheduling*, Vol. 13, No. 1, pp.17–38.