International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 12 (2018) pp. 10286-10292

Deadline Aware Multi-Objective Dragonfly Optimization Technique for Scheduling Jobs in Multi-Cluster Environment

[1]Deepanshi Nanda, [2]Amit Chhabra

[1]M. Tech Scholar, [2]Assistant Professor
Department of Computer Engineering and Technology, Guru Nanak Dev University, Amritsar, India.

Abstract

Scheduling plays a significant role in parallel computing environment to assign resources to end users. Various scheduling techniques have been designed in the literature to schedule jobs in parallel environment. However, these techniques suffer from various issues such as stuck in local optima, not support deadline constraint, poor convergence speed etc. Therefore, in this paper a novel deadline constraint aware Multi-objective dragonfly optimization is proposed. Makespan and flowtime are used to design multi-objective fitness function. Extensive experiments are done by tuning the various parameters of proposed technique. Experimental analysis reveals that the proposed technique performs better than existing job scheduling strategies.

Keywords: Scheduling, Multi-objective Dragonfly Algorithm, Makespan, Deadline, Flowtime

INTRODUCTION

Over the past few years, clusters and distributed memory multiprocessors have gained a lot of attention. Multi- cluster systems are consisting of several geographically distributed clusters provide large computation power as compared to single cluster. [1] Due to large groups being able to share the multi-cluster, job turnaround time is reduced and system utilisation becomes high. This leads to increase in job sizes by permitting jobs to use processors in several clusters concurrently i.e. to employ co-allocation [2]. Suppose in a cluster a job is waiting in the ready queue. This job may require more nodes than number of available nodes on its cluster, but in collection there may be enough nodes in the multi-cluster to accommodate the job. Then the job is called co-allocated if it was mapped onto nodes that were borrowed from other clusters. [3] Also, the execution time of the co-allocated job itself as well as that of the other co-allocated job can be affected by the amount of communication produced by each co-allocated job. [4]

Due to an exceptional increase in the number of resources in different organizations, effective algorithms are required for job scheduling. Scheduling in a heterogeneous multi-cluster environment is considered as an NP-hard problem i.e. why population-based meta-heuristics are used to obtain near optimal solutions. [5] Scheduling problems can be solved using mainly two types of algorithms:-Deterministic Algorithms and Approximate Algorithms. The latter are considered because they tend to give near optimal solutions for large-scale problems in reasonable time. The former however, are not suitable for large-scale problems as they do not guarantee to provide optimal solutions even though they take lesser time.

Multi-cluster environments are a set of connected clusters(computers) that work together to offers high-performance computing for solving large number of optimization problems. These clusters are connected via a dedicated interconnection network.

Multi-objective problems are the problems having multiple objectives, generally in a conflicting nature. [6]. So, as to resolve, Pareto optimal solution are used which signify the best trade-offs among the objectives. These optimizations can be expressed as a minimizing or maximizing each objective that is

Minimization

$$H(t) = \{H_1(t), H_2(t), \dots \dots \dots H_M(t)\} \tag{1}$$

Subject to:

$$D_i(t) \geq 0, i = 1,2,3,4, \dots \dots \dots \dots .. p \tag{2}$$

$$D_i(t) = 0, i = 1,2,3 \dots \dots \dots ... r \tag{3}$$

$$lb_i \geq t_i \geq ub_i, i = 1,2, \dots \dots \dots m \tag{4}$$

Where t is the decision vector that represent a solution; M defines total objectives, p denotes inequality constraints, with r equality constraints, and $[lb_i, ub_i]$ are the boundaries of i^{th} variable. [7]

Multi-objective Dragonfly algorithm consist of a set of non-dominated solutions which is known as pareto optimal set.

Pareto Dominance: -

$$H(t) = \{H_1(t), H_2(t), \dots \dots . H(t)\} \tag{5}$$

Consists of M objectives. Consider two solution vectors $t_1 and\ t_2$Then solution t_1 is said to dominate t_2 if it satisfied below two conditions: -

$$\forall_j \in \{1,2,3, \dots \dots . M\}: H_j(t_1) \leq H_j(t_2) \tag{6}$$

$$\exists_k \in \{1,2, \dots \dots M\}: H_k(t_1) \leq H_k(t_2) \tag{7}$$

From the above two conditions (6), (7) if any one condition is violated, the solution t_1 does not dominate the solution t_2

Pareto optimal set: A set containing Pareto optimal solutions is known as Pareto optimal set.

Pareto optimal front: - A set containing Pareto optimal solutions corresponding to objective values.

The paper is organised as follows. section 2, covers the related work. Section 3 describes MODA in a detailed manner. Experimental setup and comparative analysis of the proposed technique with existing techniques has been shown in Section 4. Finally, conclusions are presented in section 5.

RELATED WORK

Mandeep Kaur et al. (2018)[8] presented a multi-objective bacterium foraging optimization technique to schedule jobs by considering multi-objective tradeoffs among objectives functions in grid environment. An adaptive chemotaxis approach is used in MOBFOA to avoid premature convergences and to ensure that proposed algorithm converges towards pareto-optimal front.

Li Liu et al. (2017) [9] proposed a constrained genetic algorithm with adaptive penalty function for scheduling without parameter tuning in clouds. The algorithm uses crossover and mutation probabilities in order to prevent premature convergence while satisfying the deadline.

Yue-Shan Chang et.al (2017)[10] proposed an agent-based scheduling algorithm with deadline constraint to achieve the objective of minimum resource utilization and lower prediction error rate in federated cloud environment

Yongsheng et al. (2016)[11] proposed a scheduling algorithm to schedule non-linear parallel jobs in meteorological Cloud. The algorithm considers four characteristics of the jobs concurrently such as execution time, job deadlines, assigned resources, and system loads. The effectiveness and overall performance of scheduling algorithm are illustrated through simulations using Weather Research and Forecasting model.

Attiya et al. (2016) [12] proposed two choices scheduling algorithm with the objective of reducing makespan as well as load balancing. The algorithm randomly chooses a least loaded machine. for assigning job rather than one machine.

Shirin et al. (2014) [13] proposed a hybrid batch job scheduling in a grid environment that combines Genetic algorithm and particle swarm optimization to minimize makespan and flowtime. Results shows there is reduction to when compare to heuristic such min-max, discrete particle swarm optimization-min..

Ying Chang-tian et al. (2012) [14] describes scheduling of independent tasks in parallel environment with the main of minimize makespan along with energy usage as a scheduling consideration. Both algorithms utilize the techniques regarding double fitness to identify that performance.

Chitra et al. (2010) [15] presented a multi-objective evolutionary algorithms (MOEA) with the objective minimum makespan, average flow-time and maximization of reliability 0f scheduling problem in heterogeneous environment. The paper describes that MOEA provides a well pareto optimal fronts single run.

Yuming Xu et.al (2014) [16] proposed a multiple priority queues genetic algorithm (MPQGA) in a heterogeneous computing system in which a priority is assigned to each subtask and for the mapping of task to processor an earliest finish time strategy is used.

PROPOSED METHODOLOGY

Dragonfly algorithm (DA) is an multi-objective optimization technique inspired from ordinary behavior of dragonflies, ideally dependent on exploration and exploitation. The dragonfly creates sub-warms around various areas for navigating, food searching and survival from enemies which is useful for convergence towards pareto-optimal solutions and coverage of optimal solution along the objectives.

The objective of multi-response optimization approach is to find out accurate approximation of true Pareto optimal solutions with uniform coverage across all objectives. As swarms of living creature follow living instincts so dragonfly individuals must have attracted toward efficient utilization of resources. Swarming behaviour of the dragonfly are discussed by these following major factors.

For simulating dragonflies, swarming behaviour the characteristics must be mathematically demonstrated in following Section. The below table is described Nomenclature of the above equations:

V_k	Position of k^{th} neighbouring individual
P	Total number of jobs
U_K	Velocity of the k^{th} neighbouring jobs
V^-	Distraction from Enemy
V^+	Speed of current server
ΔV	Step vector
S_i	Separation
S	Separation weight
Y_i	Job scheduling
W_i	Attraction
W	Food factor
E	Inertia weight
Z_i	Allocation matrix
z	Cohesion Weight
D_i	Distraction
Y	Alignment Weight
d	enemy factor
C	Iteration number
g	Dimension of position vectors
z_1,z_2	Random number in [0,1]

1) The separation is modelled as:

$$X_i = -\sum_{K=1}^{P} V - V_K \tag{9}$$

2) Job scheduling is computed as:

$$Y_i = \frac{\sum_{K=1}^{P} U}{P} \tag{10}$$

3) The allocation matrix is computed as:

$$Z_i = \frac{\sum_{K=1}^{P} V_K}{P} - V \tag{11}$$

4) Attraction towards scheduling is calculated as:

$$W_i = V^+ - V \tag{12}$$

5) Distraction outwards an enemy (i.e. remaining resources) is quantified as follows:

$$D_i = V^- + V \tag{13}$$

From the combinations of above motions for each iteration the corrective pattern of dragonfly individual is determined. The position (V) and a step vector ΔV is used for updating the position of Dragonfly in each iteration. Following step vector, demonstrates the motion orientation for each dragonfly individual,

$$\Delta V_{c+1} = (xX_i + yY_i + zZ_i + wW_i + dD_i) + e\Delta V_C \quad (14)$$

Furthermore, the position vectors is computed as : -

$$V_{c+1} = V_c + \Delta V_{c+1} \quad (15)$$

In optimization from the use of separation, alignment, cohesion, food, and enemy factors (x, y, z, a and d), various explorative and exploitative behaviors' can be achieved. In order to improve randomness, when no neighbouring solutions dragonflies are required to fly around the search space using a random walk (Levy flight). From the following equation, the position of dragonflies can be updated: -

$$V_{C+1} = V_c + \text{Levy}(g) \times V_c \quad (16)$$

The Levy flights calculated as follows: -

$$\text{Levy}(x) = 0.01 \times \frac{z_1 \times \sigma}{|Z_2|^{\frac{1}{F}}} \quad (17)$$

F is a constant.

$$\sigma = \left(\frac{\tau(1+F) \times \sin(\frac{\pi F}{2})}{\tau(\frac{1+F}{2}) \times F \times 2(\frac{F-1}{2})}\right)^{\frac{1}{F}} \quad (18)$$

where, $\tau(x) = (x-1)!$

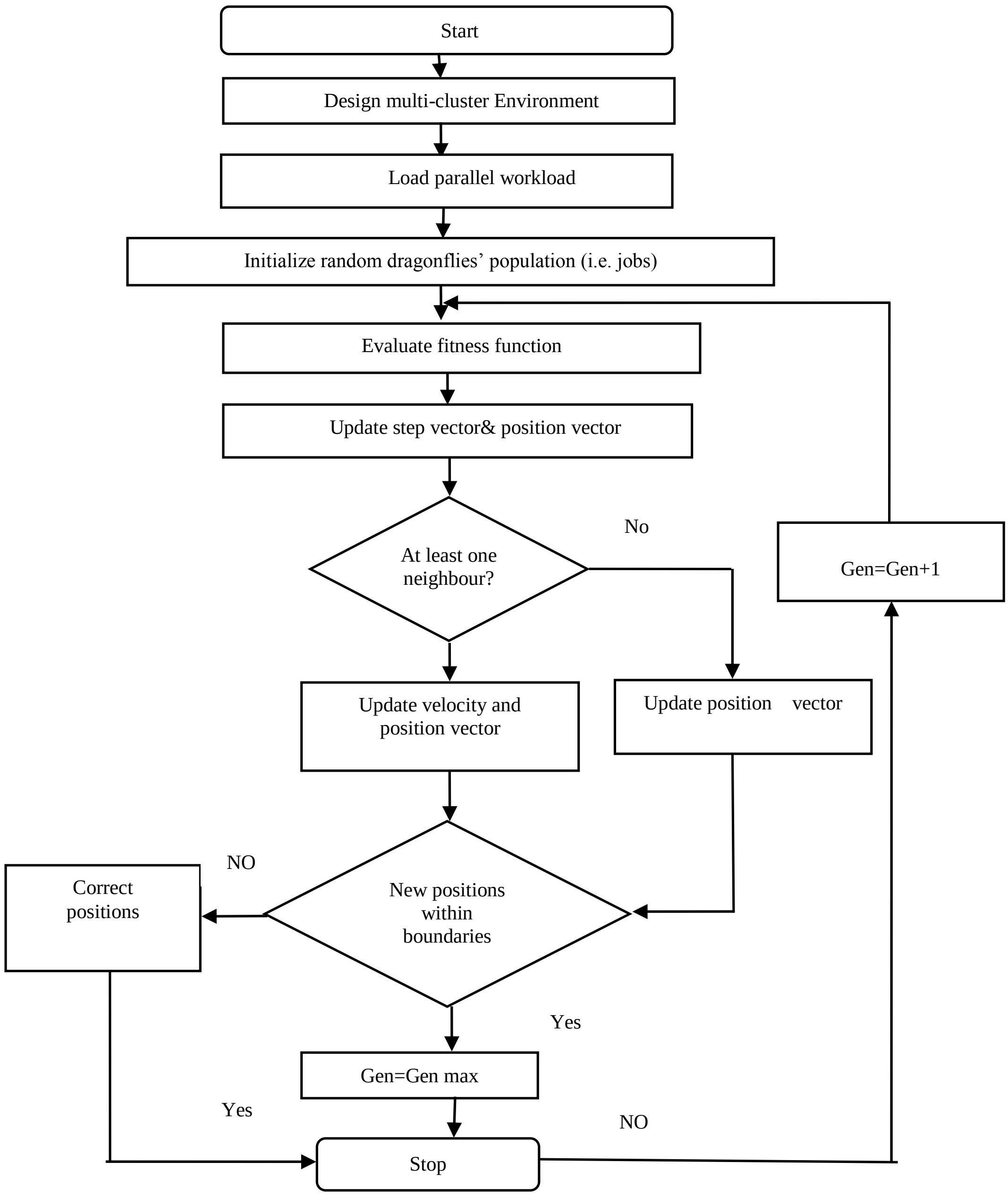

Figure.1.1: The flowchart of proposed methodology

RESULTS AND DISCUSION

This section describes the performance analysis in terms of Makespan and Flowtime. For performing experiments, a heterogeneous multi-cluster environment is created. comprising of 4 clusters. Each cluster has a varying computational speed. The experiment evaluation is done using 300,500,700 set of jobs are used.

Table 1: Overview: heterogeneous multi-cluster environment

No. of Clusters	Resources	Speed
1	32	1
2	32	2
3	32	3
4	32	4

A. Flow Time

The flow time of a schedule provides a a method of measuring of the average time period which a job requires inside a computer system as well as the average number of incomplete jobs in the system. It can also describe as the sum of the completion times of all jobs within the system.

$$Flow\ Time = \frac{Total\ processing\ time + total\ waiting\ time}{Number\ of\ jobs}$$

B. Makespan

The sum total period of time needed to complete a group of jobs is known as makespan. [17]

C. Miss rate: It is total number of jobs that missed the deadline

$$\text{Miss rate} = \frac{\text{Completed Jobs}}{\text{Total Jobs}} \times 100$$

D Deadline: -It refers to instant of time through which a job must be completed.

$$Deadline = Arrival\ time\ of\ a\ job + A * Bursttime\ of\ a\ job$$

Here A is calculated as: -

$$= rand() * (Amax - Amin) + Amin$$

Where the values of $A_{max} = 1\ or 1.5 or 2,$

$$A_{min} = 0.1$$

The performance analysis is conducted by varying the values of the parameters such as number of jobs, processor and constant parameter. Results that proposed algorithm outperforms better result than existing algorithm.

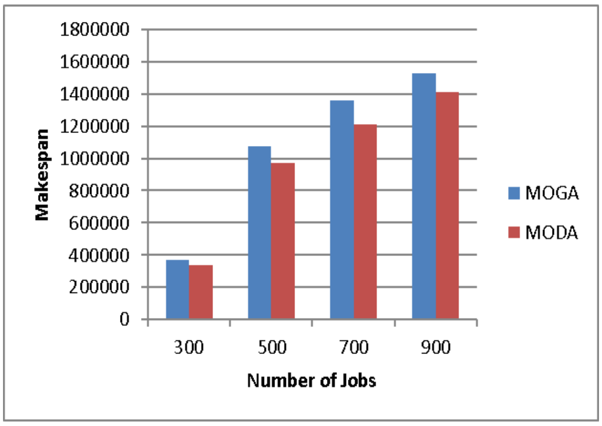

Figure 2. (a)

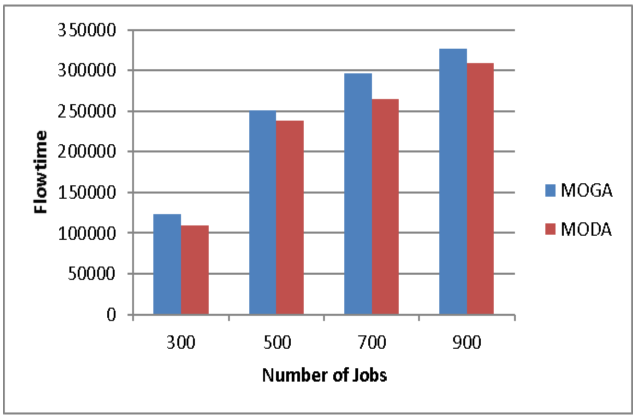

Figure 2. (b)

Figure 2(a). and Figure 2(b) indicates the comparison of MOGA and MODA in terms of makespan and flowtime by varying job size and keeping fixed processors. i.e(128)

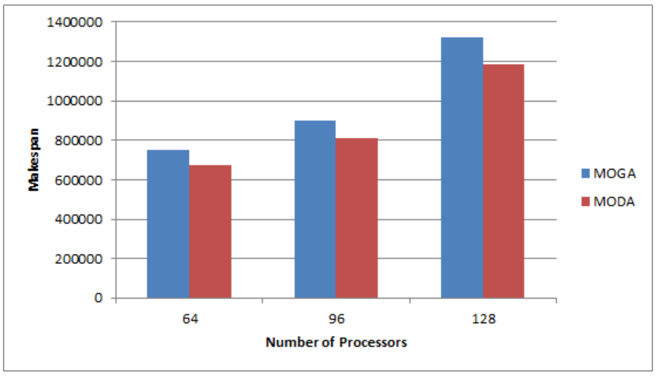

Figure 3. Comparison of makespan with 500 jobs and (128,96,64) processors

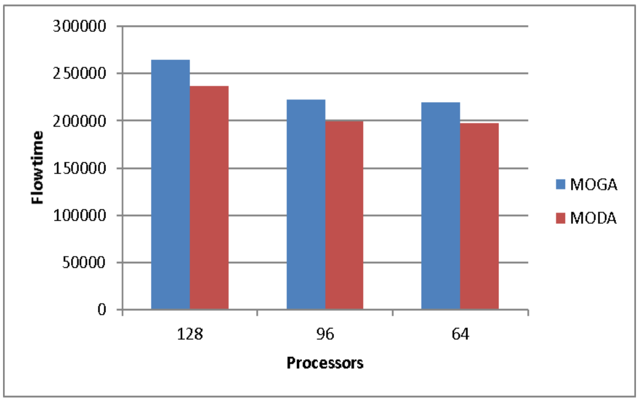

Figure 4. Comparison of Flowtime by varying number of processor.

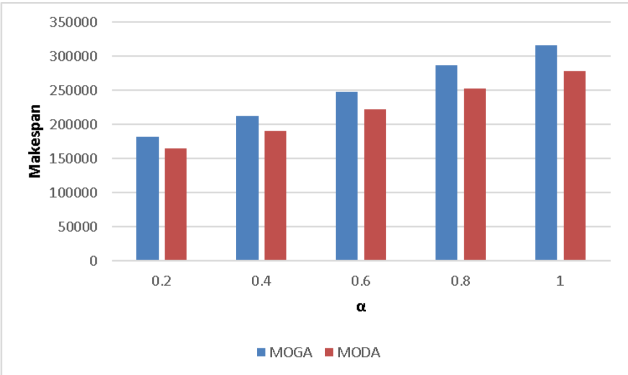

Figure 5(a)

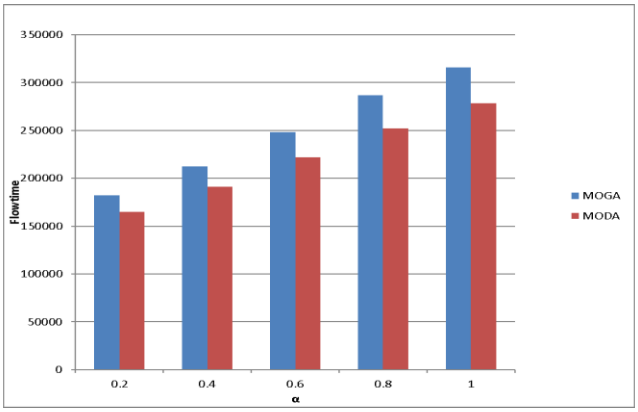

Figure 5(b)

Figure 5(a) and 5(b): depicts the comparison of Makespan and Flowtime obtained in terms of $\propto$.

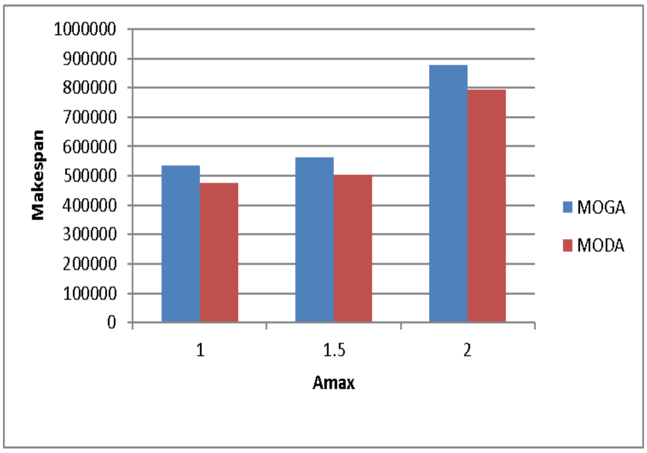

Figure 6. Comparison of MOGA and MODA in terms of makespan for 500jobs with $A_{max}(1,1.5,2)$

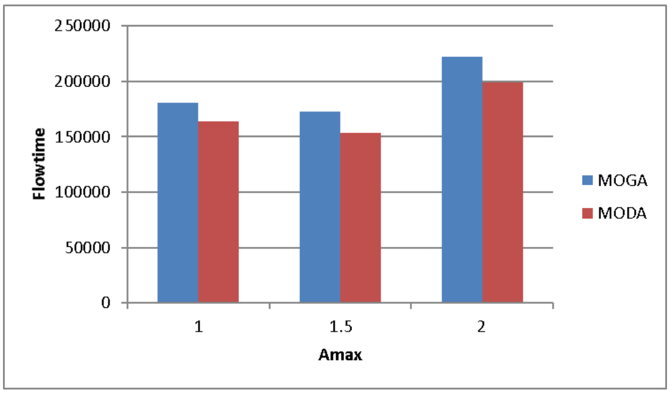

Figure 7. comparison of MOGA and MODA in terms of flowtime for 500jobs

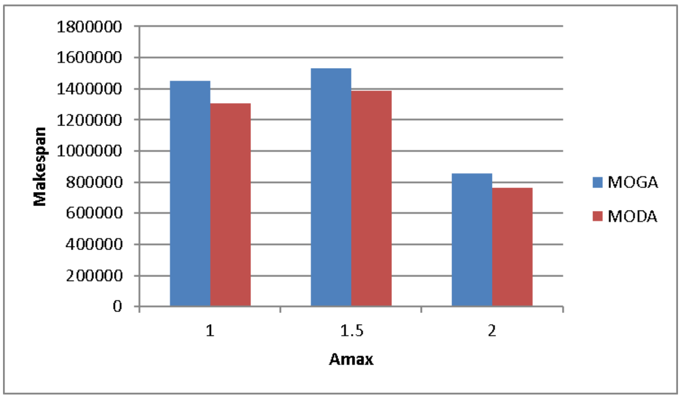

Figure 8(a)

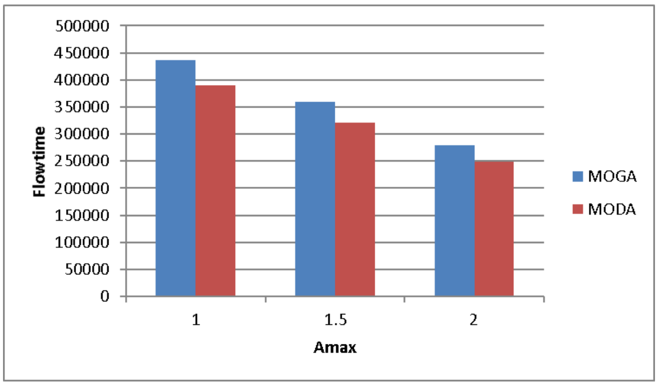

Figure 8(b)

Figure 8(a) and 8(b) indicates the comparison of makespan & flowtime in terms of A_{max} value with 900 jobs. The result shows that the $A_{max} = 1$ for deadline evaluation provides significant results compared to other A_{max} value i.e. (1.5, 2)

CONCLUSION

Extensive review of existing scheduling techniques shows that designing an efficient job scheduling technique is an ill-posed problem. The effect of deadline constraint is neglected by majority of existing researchers while designing a meta-heuristic based job scheduling technique. Therefore, in this work, a novel deadline aware job scheduling technique is proposed. Additionally, multi-objective dragonfly is used to assign resources to end users' jobs. The proposed technique has been designed and Implemented in the MATLAB 2017 toolbox with the help of wireless communication toolbox. Extensive experiments indicate that the proposed technique outperforms existing techniques in terms of makespan and flow time.

This work has not considered that effect of server failure while designing the scheduling technique. In near future, a suitable failure aware jobs scheduling technique will be designed.

REFERENCES

[1] Bawajy, Jemal H., and Sivarama P. Dandamudi. "Parallel Job Scheduling on Multicluster Computing Systems." cluster. Vol. 3. 2003.

[2] Javadi, Bahman, Mohammad K. Akbari, and Jemal H. Abawajy. "Performance analysis of heterogeneous multi-cluster systems." Parallel Processing, 2005. ICPP 2005 Workshops. International Conference Workshops on. IEEE, 2005.

[3] Yang, Chao-Tung, et al. "Well-balanced allocation strategy for multi-cluster computing environments." Future Trends of Distributed Computing Systems, 2008. FTDCS'08. 12th IEEE International Workshop on. IEEE, 2008

[4] Blanco, Héctor, et al. "Multiple job co-allocation strategy for heterogeneous multi-cluster systems based on linear programming." The Journal of Supercomputing 58.3 (2011): 394-402.

[5] Xu, D., Sun, K. and Li, H., 2008. Parallel machine scheduling with almost periodic maintenance and non-preemptive jobs to minimize makespan. Computers & operations research, 35(4), pp.1344-1349.

[6] Knowles, Joshua, and David Corne. "The pareto archived evolution strategy: A new baseline algorithmfor pareto multiobjective optimisation."In Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on, vol. 1, pp. 98-105. IEEE, 1999.

[7] MOBJECTUVE K. Deb, M. Mohan, S.Mishra, A Fast Multi-objective Evolutionary Algorithm for Finding Well-SpreadPareto-Optimal Solutions. KanGAL Report Number: 2003002, Indian Institute of Technology, Kanpur, India (2003).

[8] Kaur, Mandeep, and Sanjay Kadam. "A novel multi-objective bacterium foraging optimization algorithm (MOBFOA) for multi-objective scheduling." Applied Soft Computing 66 (2018): 183-195.

[9] Liu, Li, Miao Zhang, Rajkumar Buyya, and Qi Fan. "Deadline-constrained coevolutionary genetic algorithm for scientific workflow scheduling in cloud computing." Concurrency and Computation: Practice and Experience 29, no. 5 (2017)

[10] Chang, Y.S., Fan, C.T., Sheu, R.K., Jhu, S.R. and Yuan, S.M., 2018. An agent-based workflow scheduling mechanism with deadline constraint on hybrid cloud environment. International Journal of Communication Systems, 31(1).

[11] Hao, Yongsheng, Lina Wang, and Mai Zheng. "An adaptive algorithm for scheduling parallel jobs in meteorological Cloud." Knowledge-Based Systems 98 (2016): 226-240.

[12] Attiya, Ibrahim, Xiaotong Zhang, and Xiaoyu Yang. "TCSA: A dynamic job scheduling algorithm for computational grids." In Computer Communication and the Internet (ICCCI), 2016 IEEE International Conference on, pp. 408-412. IEEE, 2016.

[13] Zahedani, Shirin Dehghani, and GholamHossin Dastghaibyfard. "A hybrid batch job scheduling algorithm for grid environment." In Computer and Knowledge Engineering (ICCKE), 2014 4th International eConference on, pp. 763-768. IEEE, 2014.

[14] YING, Chang-tian, Jiong YU, and Xing-yao YANG. "Energy-aware task scheduling algorithms in cloud computing [J]." Microelectronics & Computer 5 (2012): 044.

[15] Chitra, P., Revathi, S., Venkatesh, P. and Rajaram, R., 2010, February. Evolutionary algorithmic approaches for solving three objectives task scheduling problem on heterogeneous systems. In Advance Computing Conference (IACC), 2010 IEEE 2nd International (pp. 38-43). IEEE

[16] Xu, Yuming, Kenli Li, Jingtong Hu, and Keqin Li. "A genetic algorithm for task scheduling on heterogeneous computing systems using multiple priority queues." Information Sciences 270 (2014): 255-287

[17] Mirjalili, S., 2016. Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Computing and Applications, 27(4), pp.1053-1073.

[18] Johannes, B., 2006. Scheduling parallel jobs to minimize the makespan. Journal of Scheduling 9(5), 433-452.