International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 12 (2018) pp. 10449-10457

Scheduling Parallel Jobs with Deadline and Makespan Constraints using Multi-Objective Monarch Butterfly Algorithm

Simrat pal kaur, Amit Chhabra

Master of computer science, G.N.D.U, Amritsar, Punjab, India.

Assistance Professor, G.N.D.U, Amritsar, Punjab, India.

Abstract

To minimize the scheduling issues in parallel computing, we describes the scheduling of independent tasks using makespan with deadline as a scheduling importance. Parallel computing facilities to scheduling on different nodes becomes an important concerns in the ongoing years. In multi-objective heterogeneous computing nodes enhances large amount of scheduling and resource allocation to optimize performance and application tasks are factor that make the scheduling on NP Hard problem. For high scheduling of parallel jobs in grid environment we used metaheuristics to find near optimal solution. In this paper, we study scheduling of parallel jobs and concentrate on high-performance workload and optimizes the parameters such as for instance deadline, makespan and flowtime with the utilization of multi-objective monarch butterfly optimization algorithm (MOMBOA) predicated on multi-objective genetic algorithm (MOGA). Our aim should be to idle for high scheduling of parallel jobs in heterogeneous environment. The end result show the potency of the proposed method MOMBO providing solutions that enhance the performance, makespan and deadline regarding other well-known techniques in the literature.

Keyword: Parallel computing, Scheduling, Monarch Butterfly Optimization Algorithm (MOMBOA).

INTRODUCTION

In computer Science there are several techniques to solve the problems either on single processor or multiprocessor. It is impossible to solve large problems on single processor with high computation power. So, first to solve large problems with high computation power at constant or reasonable computation time and second make them robust at the characteristics of the define problem are the aims of parallel metaheuristic. To solve complex and large problems of realistic scale can be offer by metaheuristic.

Multi-objective in parallel computing environment are generally used for better performance because it breakdown the workload of complex jobs into smaller ones that enhance to high performance with better makespan. Several cluster are used to develop these types of environment because in grid environment several clusters are used to interconnect network resources with performance of computation. Performance can be increased by better utilization of resources and it depends upon the execution time i.e. startup time of jobs.

In existing methodology have many limitation like ant colony based (ACB) technique is that it converges at low speed in the initial stages and requires more time and energy to converge and not guarantee is to find optimal solution this happens as a result of incorrect choice of the initial probable parameter. So, we use Monarch Buttery Optimization (MBO) in parallel computing is based upon job scheduling techniques and MBO is a population based algorithm and have better execution on multi-objective fitness function and it find the near optimal solution among the existing and proposed techniques based on the makespan, deadline and flow time.

The aim of parallel meta-heuristics, to find near and optimal solutions to increase performance together with the most approaches to instantiate them for neighborhood-and population based meta-heuristics. To solve the computation using deadline and makespan with parallel metaheuristic.

RELATED SEARCH

Crainic et al. (2017) present an over-all view of parallel meta-heuristic for optimization is based upon cooperation-based strategies, which display remarkable performances on asynchronous exchanges and the creation of new information out of exchanged data to boost the global guidance of the search.

Hedieh Sajedi et al. (2017) approves a new algorithm is established for scheduling in grid computing that provides a genetic algorithm, to reduce the completion time of machines and avoids trapping in a local minimum effectively. It has the capability of globally optimization.

Mohammad Masdari et al. (2016) provided the detailed analysis of workflow scheduling schemes for parallel computing. It focuses on giving solution to the issues that is, proper execution of workflows e.g. budget and deadline constraints.

Hang Qu et al. (2016) gave steering regarding the new programing design that allows high performance workloads to run on thousands of cores, named as Canary. Canary has been a primary that borrows distributed programing from high performance computing to execute quick codes. In Canary, the duties of the controller and employees are swapped then arranged the required knowledge and assigns knowledge partition to the controller. Every controller then schedules tasks in line with the allotted partition.

Saurabh Bilgaiyan et al. (2014) every controller schedules tasks in line with the allotted partition. Evolutionary

optimization techniques that are mostly used for scheduling and task-resource mapping on parallel computing.

Seyedali Mirjalili et al. (2014) proposed a comparative study with attractive force Search algorithmic program (GSA), Particle Swarm optimization (PSO), Differential Evolution (DE), Evolution Strategy (ES), and organic process Programming (EP).

Wang et al. (2014) In MBO Monarch butterflies can be placed in two places i.e. northern USA and southern (place 1) and Mexico (place 2). Initially, by migration operator the offspring's are produced, which can be balanced butterfly adjusting operator, which can be balanced by the migration operator. It is taken after by tuning the situations for different butterflies by methods for butterfly adjusting operator. **Yiqiu Fang et al. (2011**) introduce task scheduling in two levels to balance the load in an efficient manner. The two level scheduling mechanism increases the utilization of resources.

Ying Chang-tian et al. (2013) Dynamic voltage scaling technique is used to minimization of power usage and genetic algorithm is used for selecting best scheduling critirea for both algorithms utilize the techniques and double fitness to identify that performance and select individuals also undertake that genetic algorithmic program to duplicate consider the reasonable scheduling technique. The resources are configured then energy consumption reduces by parallel applications to computation nodes.

Eloi Gabaldon et.al (2017). The analysis in programing has targeting not solely optimizing the energy consumed by the processors however additionally optimizing the makespan, i.e., job completion time. The massive range of heterogeneous computing nodes.

Mahfouz Alam et al. (2016) focused on total workload is divided according to the processors of the distributed system to progress resource utilization and its task's response time; it should ignore a condition in which few processors are under loaded or overloaded or moderately loaded.

Fahimeh Farahnakian et al. (2015) have presented architecture of distributed systems to execute a VM Consolidation dynamically to minimize power usage for data centers of parallel. It introduced Ant Colony system based VM Consolidation (ACSVMC) methodology finds out the near-optimal solution with different particular unbiased function .Experiment outcomes upon real data indicate that ACSVMC minimizes power usage and also keeps the required efficiency in a parallel data centers.

Javid Taheri et al. (2014) works on scheduling phase and provide two algorithms to reduce the make-span for executing all jobs and transfer time for all data-fields. It use two collaborating algorithm for schedule job & replicate data-fields to connected nodes and storage nodes for large system; it can address the bulk scheduling mode.

T. Vigneswari et al. (2014) proposed job programming for economical utilization of grid resources. Programming of jobs is difficult and NP complete.

Sukhpal et al. (2014) presented K-means clustering algorithm has been used to analyze and cluster workloads for assigning weights. In a parallel environment, needs a challenge to effectively perform scheduling process. The distinguishing work done was creation of workload used to categorize different types of workloads and then schedule them. Clustering of workloads is then done by k-means based clustering algorithm. As a result effective resource management and scheduling is done.

Joanna Kołodziej et al. (2013) have tackled the particular 01independent batch scheduling within the computational grid like a bi-objective global reduction issue with makespan and power usage as primary requirements and also applied Dynamic voltage Scaling Frequency technique towards energy used by the particular grid resources.

Jing Liu et al. (2013) introduced model for scheduling, the resolution solution dependent on multi-objective genetic algorithmic rule (MOGA) is created together crossover operators, choice operators and approach to selecting solutions. It possesses a balance for general performance associated with numerous objects.

Ying Chang et al. (2012) describes scheduling of independent tasks as minimization issue with in parallel computing using makespan along with energy usage as a scheduling considerations. These algorithms utilize the techniques regarding unite and fitness to identify that performance and select individuals its results indicates choose the best agreement in between makespan and power usage.

Sid Ahmed et al. (2010) On heterogeneous distributed resources computational grids have the potential for resolution large-scale scientific issues. The key technical hurdles should overcome before this potential may be completed. To effective utilization of machine grids is that the efficient co-allocation of jobs.

METAHEURISTICS

Multi-objective Genetic Algorithm is used as existing and Multi-objective Monarch Butterfly Optimization Algorithm is our proposed methodology that are used for new research.

A. Genetic algorithm

Genetic Algorithm is a metaheuristic inspired optimization method based on natural selection of population. For optimization Genetic algorithms (GA) is widely to produce high-quality solutions by using mutation, crossover and selection operators. The essential steps for genetic algorithmic are:

a. To start algorithm **initial population is** required; population is a set of random strings that can be generate by random generator. Fitness function is identified for each population string.

b. The **crossover operator** picks sets of strings aimlessly and delivers new matches.

c. The **mutation operator** unpredictably transforms or turns around the estimations of bits amid a string. The amount of change activities is set by a mutation rate.

1. *Decide initial population of people.*
2. *Judge the fitness perform by the people.*
3. *Repeat*
4. *The most effective people to be selected.*
5. *Generate new people by applying crossover and mutation operator.*
6. *The fitness perform for brand spanking new people ought to be examined.*
7. *The worst people square replaced with the best ones.*
8. *Till a shopping criteria is met.*

Figure 1: use to describe the phases of GA.

B. Monarch Butterfly Algorithm

Monarch butterfly has an orange and black colour. Male and female monarch have different wings to recognize them. It is a milkweed margarine x in the family Nymphalidae. The eastern North American monarch is known for its capacity of moving by flying a large number of miles from Place 1 (USA and southern Canada) to Place 2 (Mexico) each late spring; flying over west of the Rough Mountains to California. The female ones lay eggs for producing population and a few butterflies perform Levy flight when they move.

Monarch butterfly optimization

With a specific end goal to make the migration behaviour of monarch butterflies there are different improvement issues:

1. All the monarch butterflies are just situated in Land 1 or then move to Place 2; make up the entire monarch butterfly populace.
2. Every type monarch butterfly individual is produced by relocation administrator from monarch butterfly in Place 1 or in Place 2.
3. With a specific end goal to increase population, an old monarch butterfly will pass away once a youngster is produced. The recently produced one is at risk to be disposed of on the off chance that it doesn't display better wellness with regard to its parent. Under this situation, the parent is kept in place and undestroyed.
4. The monarch butterfly people with the best wellness moves consequently to the people to come, and they can't be changed by any administrators. This can ensure that the quality or the viability of the monarch butterfly populace will never break down with the augmentation of ages.

Monarch Butterfly Method

MBO achieve the ideal execution by and large and sexually transmitted on certain experiments. MBO technique is given.

1. Monarch Butterfly Migration Operator
2. Monarch Butterfly Adjusting Operator

Monarch Butterfly Migration Operator

Monarch butterflies (MB) can move from Place 1 called Subpopulation 1 to Place 2 called Subpopulation 2. The quantity of is $ceil(p * NP)(NP1) and\ NP - NP1\ (NP2)$.

Here, $ceil(x)$ rounds x; NP is the number of the population; p is the proportion of MB in Place 1. This relocation procedure can be as takes after.

$$x_{a,n}^{s+1} = x_{m1,n}^{s} \qquad \text{eq. (1)}$$

Where $x_{a,n}^{s+1}$ is the s^{th} component of x_a generation $s+1$ is the situation of MB a and $x_{m,,n}^{s}$ is the s^{th} component of x_m that is the MB m_1 recently produced position of the randomly choose from Subpopulation 1. When $m \le p$, the component n in the recently created MB is created by Eq. (1). Here,

$$m = rad * pri \qquad \text{eq. (2)}$$

pri is movement period and rad is an arbitrary number. If $m > p$, the component n in the recently created MB is produced by

$$x_{a,n}^{s+1} = x_{m2,n}^{s} \qquad \text{eq. (3)}$$

Where, $x_{m2,n}^{s}$ is the s^{th} component of x_{m2} is randomly chosen and recently created position of MB from Subpopulation 2. From Subpopulation 1 MBO strategy can adjust the movement administrator by proportion p. The movement administrator can be spoken to in Algorithm 1.

Algorithm* 1 : *Migration operator

Start

for $a = 1\ to\ NP_1$ **do**

// (*for all MB in Subpopulation* 1)

for $n = 1\ to\ D$ **do**

//(*for every component* a^{th} *MB*)

Randomly create rad by uniform distribution;

$m = rad * pri$;

if $rad \le p$ **then**

Randomly select MB in Subpopulation 1 (m_1);

Create n^{th} *componenet of the* x_a^{s+1} *as eq.* (1).

else

Randomly select MB in Subpopulation m_2);

Create n^{th} *componenet of the* x_a^{s+1} *as eq.* (3).

End if

End *for n*

End *for a*

End.

Monarch Butterfly Adjusting Operator

The procedure of butterfly modifying administrator can be basically portrayed as takes after.

For every one of the components in MB b, if a haphazardly created number rand is littler than or equivalent to p, it can be refreshed as

$$x_{b,n}^{s+1} = x_{best,n}^{s} \qquad \text{eq. (4)}$$

where $x_{b,n}^{s+1}$ is the n^{th} component of x_b at age of population $s+1$ that exhibits the situation of the MB b. So, $x_{best,n}^{s}$ is the n^{th} component of x_{best} that is the $best$ MB in Place 1 and Place 2. n is present age number. On the differentiation, if $rand$ is greater than p, it can be updated as

$$x_{b,n}^{s+1} = x_{m3,n}^{s} \qquad eq.(5)$$

Where $x_{m3,n}^{s}$ is the kth component of x_{m3} that is arbitrarily chosen in Place 2.

Here, $x_{m3} \in \{1,2, \dots \dots, NP_2\}$. Under this condition, if $rand > BAR$, it can be additionally refreshed as takes after.

$$x_{b,n}^{s+1} = x_{b,n}^{s} + \alpha * (d_{xn}) - 0.5 \quad \text{Eq. (6)}$$

Where BAR demonstrates butterfly changing rate. dx is the walk venture of the MB b that can be figured by performing Le'vy flight.

$$dx = Levy(x_b^s) \qquad \text{Eq. (7)}$$

In Eq. (6), α is the weighting factor that is given as Eq. (8)

$$\alpha = S_{max}/t^2 \qquad \text{Eq. (8)}$$

Where S_{max} is walk step can move in one stage, and t is the present age. Monarch butterfly administration can be in algorithm 2.

Algorithm 2 : Adjusting Butterfly operator

Start
for $b = 1$ *to* NP_2 **do**
//(*for all MB in Subpopulation* 2)
Calculate the walk step d_x *by Eq.* (7);
Figuring the weighting factor by eq. (8);
for $n = 1$ *to* D **do**
//(*every element in* b^{th} *MB*)
Randomly produce a rad by uniform dispersion;
if $rad \leq p$ **then**
C*reate* n^{th} *componenet of the* x_b^{s+1} *as eq.* (4).
else
Randomly select a MB in Subpopulation 2(m_3);
Create the n^{th} *component of the* x_b^{s+1} *as eq.* (5).
if $rand > BAR$ **then**
$x_{b,n}^{s+1} = x_{b,n}^{s+1} + \omega * (d_n - 0.5)$;
End if
End if
End *for n*
End *for b*
End.

Advantages of MBO

1. It has awesome in hence on its execution.
2. Best parameter will be chosen with the utilization of MBO
3. Computational prerequisites are of essential significance for any metaheuristic strategy. It is basic to enhance the pursuit speed by dissecting the MBO strategy.
4. Genuine applications, ought to be utilized for effective execution of the MBO technique, for example, picture division, compelled enhancement, rucksack issue, booking, dynamic advancement, radio wire and microwave outline issues, and water, geotechnical and transport building.
5. The qualities of the relocation conduct (basically movement administrator and margarine x altering administrator) are romanticized to frame the MBO strategy.
6. More qualities, for example, swarm, resistance against predators, and human interactions, can be admired and simplied to be added to the MBO technique.
7. Efforts ought to be made to enhance the normal execution by refreshing the hunt process.
8. MBO strategy managing high-dimensional capacities and for the low dimensional capacities, MBO performs more regrettable.

PROBLEM DESCRIPTION AND PROPOSED WORK

Grid computing has become very popular due to its wide range of applications via internet. The service composition based techniques that are conscious from the server selection from the cloud can progress to the cost and efficiency of parallel computing. This thesis has focused on the different Genetic Algorithms (GA) based makespan, deadline and flowtime techniques.

The main limitation of the ant colony based technique is that it converges at low speed in the initial stages and requires more time and energy to converge. This happens as a result of incorrect choice of the initial probable parameter. Therefore, in this work Monarch Buttery Optimization based job scheduling technique will be used. Because, it not suffers from poor convergence speed, pre-mature convergence and not stuck in local optima issue is the main motivation behind this research work.

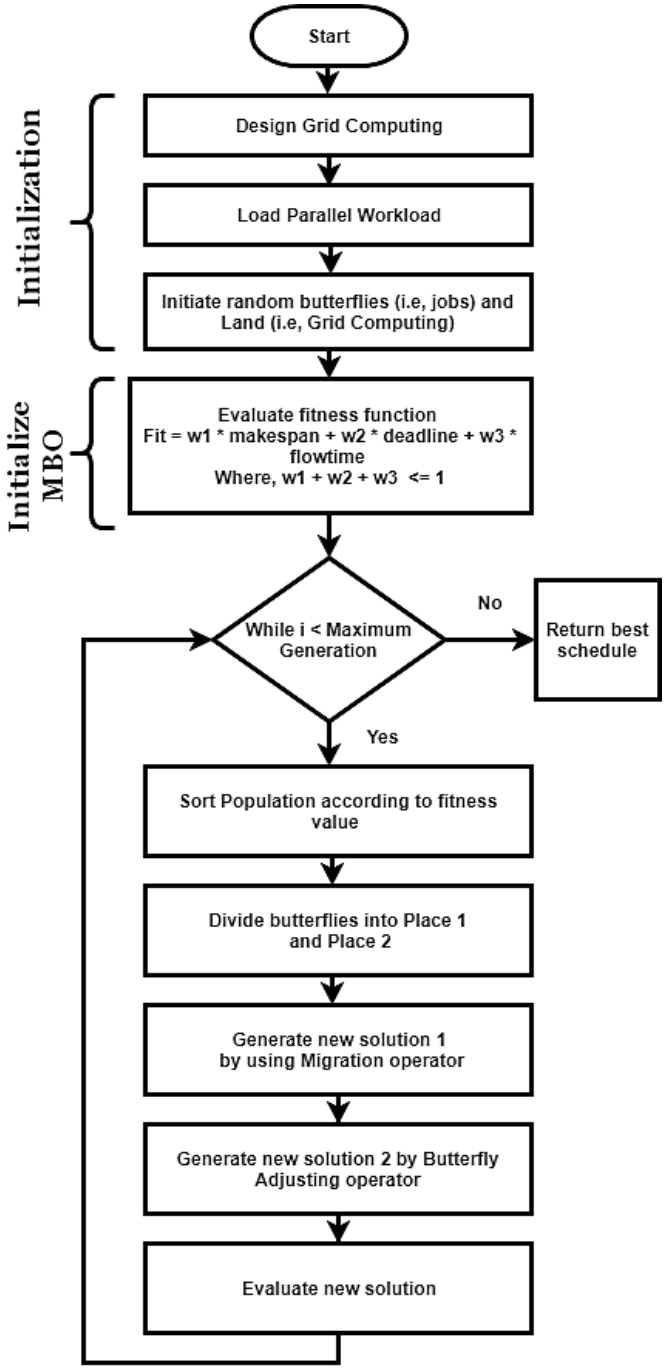

Figure 1: Proposed Methodology of monarch Butterfly Algorithm

OBJECTIVES

Main objectives are as follows:

1. To evaluate the performance of existing meta-heuristics in parallel computing is based upon job scheduling techniques.
2. To propose Monarch Buttery Optimization in parallel computing is based upon job scheduling techniques.
3. To design multi-objective fitness function which will use makespan, deadline and flow time.
4. To draw comparison among the existing and proposed techniques based on the makespan, deadline and flow time.

EXPERIMENTS

In experiment section we study about simulation. The simulations are done in Matlab environment. The makespan, flow time and deadline are considered in simulations.

According to experimental part of the research we create heterogeneous multi-cluster environment and on it scheduling parallel jobs that simulate workflows execution on the set of resources that have been taken as 64, 96 and 128 as shown in Table 1 and 300, 500, 700 and 900 jobs set is used. Each job is executed on computational resource at a completion of time. Every computational resource has predefined value. For MOMBO we used parameters such as: mutation probability – 0.8, crossover probability – 0.6, population size – 50, populations' interactions number – 130, iterations number – 20 and α – 0.5

Table 1: Overview of Multi-cluster environment

Cluster	Resources	Speed
1	64	1
2	16	2
3	16	3
4	32	4

Performance Matrices: below are some performance metrics that are desired by different types of users and providers:

A. Makespan: Makespan refers to the total length of the schedule i.e. the finishing time of the last task. İt is the most popular optimization criterion and indicates the productivity of a system. Lesser the value of makespan, more efficient is the scheduler.

$$Makespan = Max(flowtime)$$

Where t_k is the finishing time of task k.

B. Flowtime: The sum of finalization times of all tasks is called flow time.

$$Flowtime = \sum k \in tasks\ (t_k)$$

Where t_k is the finishing time of task k.

First Experiment: In this experiment we have fixed all set of resources and alpha then varies set of jobs as 300,500,700 and 900 jobs and find the fitness value of makespan and flowtime with deadline constraint .as shown in table 2.

Table 2: Makespan and Flow time with different number of jobs

Jobs	Makespan MOGA	Makespan MOMBO	Flowtime MOGA	Flow time MOMBO
300	681250	757968	195688	175965
500	1048786	1166679	329132	296484
700	1473463	1638073	456317	411508
900	1864757	2070233	554591	497605

Second Experiment: we have fixed all set of jobs and alpha and varies consisted of four instances of resources with corresponding 64, 96 and 128 configuration and find the fitness value of makespan and flowtime with deadline constraint as shown in table 3.

Table 3: Makespan and Flow time with different configuration of resources

Resource	Flowtime MOGA	Makespan MOGA	Flowtime MOMBO	Makespan MOMBO
64	247932	865016	223895	781392
96	305600	1075113	272836	957802
128	332098	1479489	298940	1330309

Third Experiment: we find the fitness value of makespan and flowtime with and without deadline constraint as shown in table 4.

Table 4: Makespan and Flow time with (1st row) and without (2nd row) deadline constraint

Flowtime MOGA	Makespan MOGA	Flowtime MOMBO	Makespan MOMBO
378550	1571910	340571	1405768
293696	1049685	262022	942825

Fourth Experiment: we have fixed jobs and resource configuration and find the fitness value of makespan and flowtime at every predefined value by varying the values of alpha as 0.2, 0.4, 0.6 and 0.8 as shown in tale 5.

Table 5: Makespan and Flow time with different alpha values.

α	Makespan MOGA	Makespan MOMBO	Flowtime MOGA	Flowtime MOMBO
0.2	1571910	1405768	378550	340571
0.4	1049685	942825	293696	262022
0.6	966980	1084366	264709	262606
o.8	1233407	1103355	313316	281846
1	1422291	1264957	347792	311098

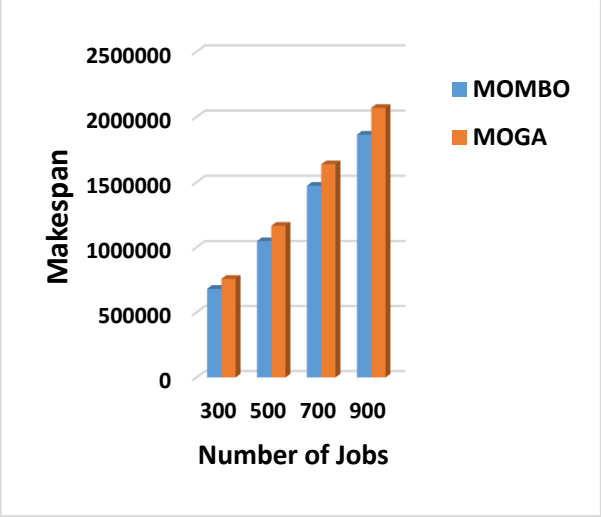

Figure 1: Makespan with different number of jobs

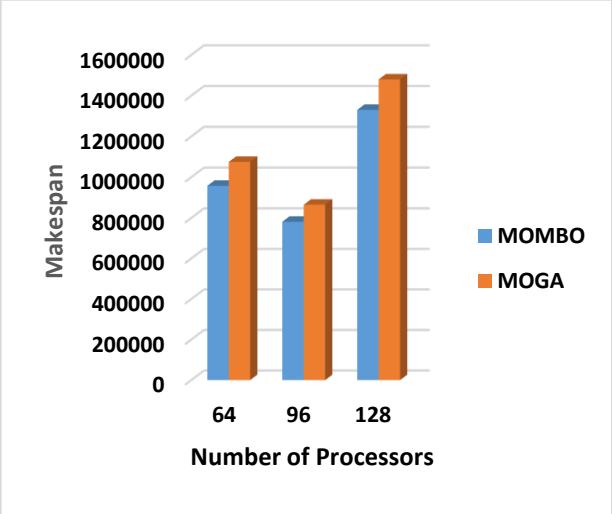

Figure 3: Makespan with different processors

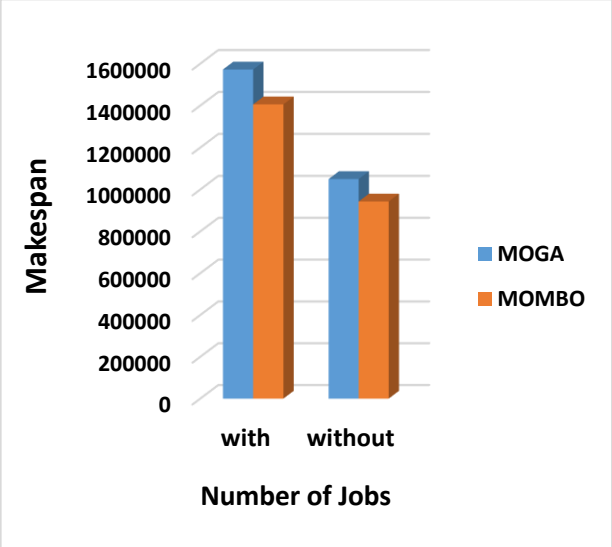

Figure 5: comparision of MOGA and MOMBO with and without deadline constraint

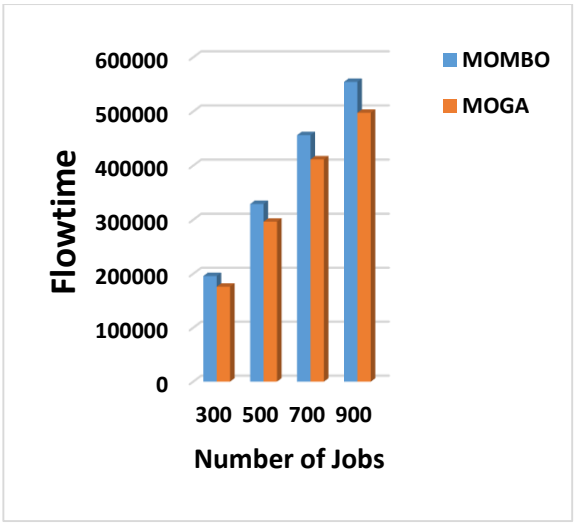

Figure 2: Flowtime with different number of jobs

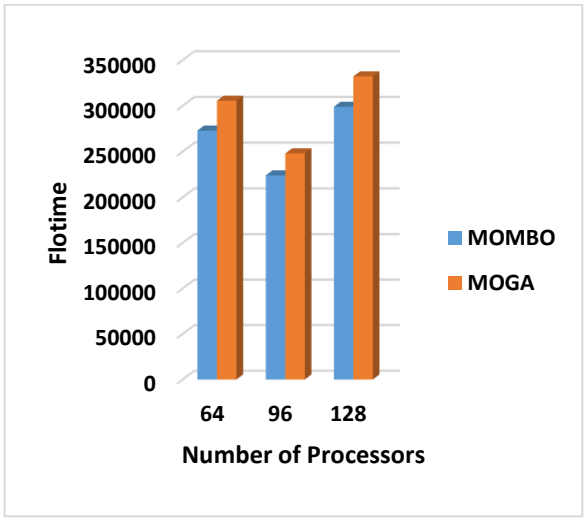

Figure 3: Flowtime with different processors

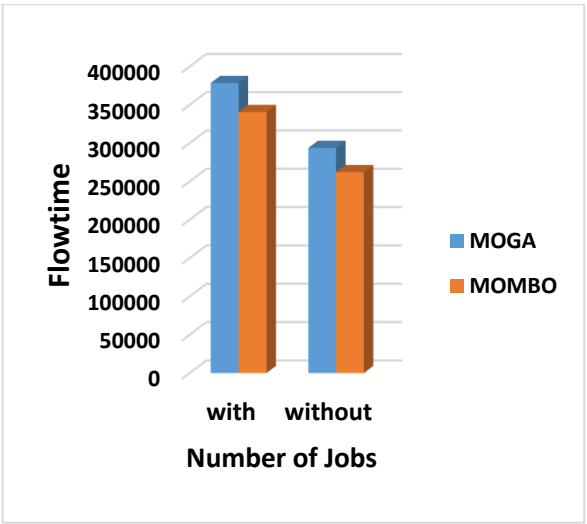

Figure 5: comparision of MOGA and MOMBO with and without deadline constraint

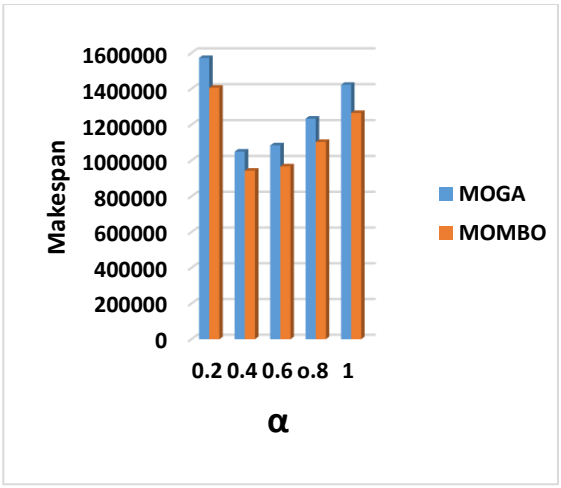

Figure 7: Makespan with vary alpha

Fig 1 compares the results of MOGA and MOMBO by varies the number of jobs (300, 500, 700 and 900) and fix α and processors on 128 to find makespan. Results clearly shows that MOMBO have better performance than MOGA and fig 2 shows the flowtime parameter with same trend.

Fig 3 compares the results of MOGA and MOMBO by varies the number of processors(64, 96, 128) and fix number of jobs at 500 and α for makespan. Results clearly shows that MOMBO have better performance than MOGA and fig 4 shows the flowtime parameter with same trend.

Fig 5 compares the results of MOGA and MOMBO to find makespan with dealine and without deadline. Results clearly shows that by using deadline constraint have better performance . Fig 6 shows the flowtime parameter with same trend.

Fig 7 compares the results of MOGA and MOMBO by varies the α (0.2, 0.4, 0.6, 0.8, 1) and fix number of jobs at 500 and processors on 128 to find makespan. Results clearly shows that MOMBO have better performance than MOGA and fig 8 shows the flowtime parameter with same trend.

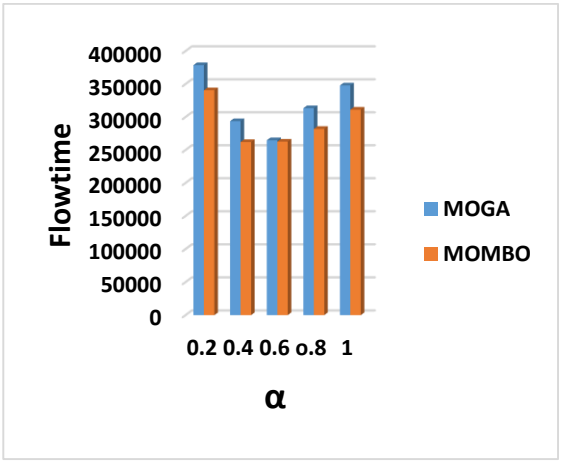

Figure 8: Flowtime with vary alpha

CONCLUSION

In this work we investigated the applicability of metaheuristic algorithms with makespan, deadline and flowtime parameters. So, for MOGA and MOMBO which can scheduling several workflows with possible deadlines constrains in heterogeneous parallel computational on matlab environment. Experimental results show the comparison efficiency of proposed MOMBO algorithm with MOGA have better much performance from the point of keeping deadlines and from the point of generating solutions with lower overall makespan and flowtime. We find the result in four steps: First, number of processors can fix with alpha and varies number of jobs. Second, fix number of jobs with alpha and varies number of processors. Third, alpha varies and constant number of processor and jobs and fourth, compare the results of flowtime and makespan with and without deadline constraint. In our future work we are going to make detailed research and get explanations of these phenomena.

REFERENCES

[1] Ahmad, Awais, Anand Paul, Sadia Din, M. Mazhar Rathore, Gyu Sang Choi, and Gwanggil Jeon. "Multilevel data processing using parallel algorithms for analyzing big data in high-performance computing." *International Journal of Parallel Programming* (2017): 1-20.

[2] Alkhanak, Ehab Nabiel, Sai Peck Lee, and Saif Ur Rehman Khan. "Cost-aware challenges for workflow scheduling approaches in cloud computing environments: Taxonomy and opportunities." Future Generation Computer Systems 50 (2015): 3-21

[3] Arsuaga-Ríos, María, and Miguel A. Vega-Rodríguez. "Energy optimization for task scheduling in distributed systems by an Artificial Bee Colony approach." Nature and Biologically Inspired Computing (NaBIC), 2014 Sixth World Congress on. IEEE, 2014.

[4] Azimi, Rasool, Mohadeseh Ghayekhloo, Mahmoud Ghofrani, and Hedieh Sajedi. "A novel clustering algorithm based on data transformation approaches." *Expert Systems with Applications* 76 (2017): 59-70.

[5] Banerjee, Soumya, and Joshua P. Hecker. "A Multi-Agent System Approach to Load-Balancing and Resource Allocation for Distributed Computing." In *First Complex Systems Digital Campus World E-Conference 2015*, pp. 41-54. Springer, Cham, 2017.

[6] Bilgaiyan, Saurabh, Santwana Sagnika, and Madhabananda Das. "An analysis of task scheduling in cloud computing using evolutionary and swarm-based algorithms." International Journal of Computer Applications 89.2 (2014).

[7] Bokhari, M. U., Mahfooz Alam, and Faraz Hasan. "Performance analysis of dynamic load balancing algorithm for multiprocessor interconnection network." Perspectives in Science 8 (2016): 564-566.

[8] Brelsford, David Paul, Waiman Chan, Alexander Druyan, and Joseph F. Skovira. "System to improve cluster machine processing and associated methods." U.S. Patent 9,723,070, issued August 1, 2017.

[9] Civicioglu, Pinar, and Erkan Besdok. "A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms." Artificial intelligence review 39.4 (2013): 315-346.

[10] Cortés-Arcos, Tomás, José L. Bernal-Agustín, Rodolfo Dufo-López, Juan M. Lujano-Rojas, and Javier Contreras. "Multi-objective demand response to real-time prices (RTP) using a task scheduling methodology." *Energy* 138 (2017): 19-31.

[11] Crainic, Teodor Gabriel. "Parallel Meta-Heuristic and Cooperative Search." (2017).

[12] Deldari, Arash, Mahmoud Naghibzadeh, and Saeid Abrishami. "CCA: a deadline-constrained workflow scheduling algorithm for multicore resources on the cloud." *The journal of Supercomputing* 73, no. 2 (2017): 756-781.

[13] Dorronsoro, B., Nesmachnow, S., Taheri, J., Zomaya, A. Y., Talbi, E. G., & Bouvry, P. (2014). A hierarchical approach for energy-efficient scheduling of large workloads in multicore distributed systems. Sustainable Computing Informatics and Systems, 4(4), 252-261.

[14] Farahnakian, F., Ashraf, A., Pahikkala, T., Liljeberg, P., Plosila, J., Porres, I., & Tenhunen, H. (2015). Using ant colony system to consolidate vms for green cloud computing. IEEE Transactions on Services Computing, 8(2), 187-198.

[15] Fang, Yiqiu, Fei Wang, and Junwei Ge. "A task scheduling algorithm based on load balancing in cloud computing." International Conference on Web Information Systems and Mining. Springer Berlin Heidelberg, 2010.

[16] Ferrandi, F., Lanzi, P. L., Pilato, C., Sciuto, D., & Tumeo, A. (2010). Ant colony heuristic for mapping and scheduling tasks and communications on heterogeneous embedded systems. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 29(6), 911-924.

[17] Gabaldon, Eloi, et al. "Blacklist muti-objective genetic algorithm for energy saving in heterogeneous environments." *The Journal of Supercomputing* 73.1 (2017): 354-369.

[18] Geist, Al, and Daniel A. Reed. "A survey of high-performance computing scaling challenges." *The International Journal of High Performance Computing Applications* 31, no. 1 (2017): 104-113.

[19] Guzek, M., Pecero, J. E., Dorronsoro, B., Bouvry, P., & Khan, S. U. (2010, June). A cellular genetic algorithm for scheduling applications and energy-

aware communication optimization. In High Performance Computing and Simulation (HPCS), 2010 International Conference on (pp. 241-248). IEEE.

[20] Kousalya, G., P. Balakrishnan, and C. Pethuru Raj. "Workflow Scheduling Algorithms and Approaches." In *Automated Workflow Scheduling in Self-Adaptive Clouds*, pp. 65-83. Springer, Cham, 2017.

[21] HOU, You-hua, et al. "Analysis on active power fluctuation characteristics of large-scale grid-connected wind farm and generation scheduling simulation under different capacity power injected from wind farms into power grid ." Power System Technology 5 (2010): 013.

[22] Kaur, Sawinder, Manojit Ghose, and Aryabartta Sahu. "Energy Efficient Scheduling of Real-Time Tasks in Cloud Environment." In *High Performance Computing and Communications; IEEE 15th International Conference on Smart City; IEEE 3rd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), 2017 IEEE 19th International Conference on*, pp. 178-185. IEEE, 2017.

[23] Luo, Guofu, Xiaoyu Wen, Hao Li, Wuyi Ming, and Guizhong Xie. "An effective multi-objective genetic algorithm based on immune principle and external archive for multi-objective integrated process planning and scheduling." *The International Journal of Advanced Manufacturing Technology*91, no. 9-12 (2017): 3145-3158.

[24] Makhlouf, Sid Ahmed, and Belabbas Yagoubi. "Resources Co-allocation Strategies in Grid Computing." *CIIA*. 2011.

[25] Masdari, M., ValiKardan, S., Shahi, Z., & Azar, S. I. (2016). Towards workflow scheduling in cloud computing: A comprehensive analysis. Journal of Network and Computer Applications, 66, 64-82.

[26] Mehra, Vishal, and Amit Chhabra. "Comparative Study of Multi Objective Task Scheduling using Metaheuristics in Multi-Cluster Environment." *International Journal of Advanced Research in Computer Science* 8, no. 5 (2017): 2648-2652.

[27] Mirjalili, Seyedali, Seyed Mohammad Mirjalili, and Andrew Lewis. "Grey wolf optimizer." *Advances in Engineering Software* 69 (2014): 46-61.

[28] Moganarangan, N., Babukarthik, R. G., Bhuvaneswari, S., Basha, M. S., & Dhavachelvan, P. (2016). A novel algorithm for reducing energy-consumption in cloud computing environment: Web service computing approach. Journal of King Saud University-Computer and Information Sciences, 28(1), 55-67.

[29] Pinel, F., Dorronsoro, B., Pecero, J. E., Bouvry, P., & Khan, S. U. (2013). A two-phase heuristic for the energy-efficient scheduling of independent tasks on computational grids. Cluster Computing, 16(3), 421-433.

[30] Qu, H., Mashayekhi, O., Terei, D., & Levis, P. (2016). Canary: A Scheduling Architecture for High Performance Cloud Computing. arXiv preprint arXiv:1602.01412.

[31] Roeva, Olympia. "Application of Artificial Bee Colony Algorithm for Model Parameter Identification." In *Innovative Computing, Optimization and Its Applications*, pp. 285-303. Springer, Cham, 2018.

[32] Sheikh, S., and A. Nagaraju. "A Comparative Study of Task Scheduling and Load Balancing Techniques with MCT using ETC on Computational Grids." *Indian Journal of Science and Technology* 10, no. 32 (2017).

[33] Singh, Sukhpal, and Inderveer Chana. "Energy based efficient resource scheduling: a step towards green computing." Int J Energy Inf Commun 5.2 (2014): 35-52

[34] Tyagi, Rinki, and Santosh Kumar Gupta. "A Survey on Scheduling Algorithms for Parallel and Distributed Systems." In *Silicon Photonics & High Performance Computing*, pp. 51-64. Springer, Singapore, 2018.

[35] Wang, Gai-Ge, Suash Deb, Xinchao Zhao, and Zhihua Cui. "A new monarch butterfly optimization with an improved crossover operator." *Operational Research* (2016): 1-25

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

[37] Younis, Muhanad Tahrir, and Shengxiang Yang. "A genetic algorithm for independent job scheduling in grid computing." (2017)

[38] Zhao, Jianfeng, and Hongze Qiu. "Genetic algorithm and ant colony algorithm based Energy-Efficient Task Scheduling." Information Science and Technology (ICIST), 2013 International Conference on. IEEE, 2013.