Received: August 21, 2018

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

OLOA: Based Task Scheduling in Heterogeneous Clouds

Pradeep Krishnadoss[1]* **Prem Jacob[2]**

[1]*St. Joseph's College of Engineering, Chennai, Tamilnadu, India*
[1, 2]*Sathyabama Institute of Science and Technology, Chennai, Tamilnadu, India*
*Corresponding author's Email: pradeepkrishnadoss@gmail.com

Abstract: Cloud computing provides increased performance, scalability and is both cost efficient as well as low accounts for maintenance, which makes it a preferred choice when the dynamic allocation of resources is considered. Of the various advantages that cloud computing provides, task scheduling is an essential feature that helps to boost performance and reduce operation cost. In the proposed OLOA, a solution is provided for optimization, taking the makespan and cost as major constraints. This is accomplished using the two algorithms, Lion optimization algorithm (LOA) and the Opposition Based Learning (OBL) algorithm; and creating a hybrid Oppositional Lion optimization algorithm (OLOA). The given solution is simulated and demonstrated in the cloudsim programming environment, where the obtained results show drastic improvement in performance, in comparison to that of the previously used other existing algorithms such as Particle Swarm Optimization (PSO) algorithm, oppositional learning based grey wolf optimizer (OGWO) and the Genetic algorithm (GA), all of which do not match the performance rates of the proposed hybrid algorithm.

Keywords: Task scheduling, Lion optimization algorithm, Oppositional based learning, Makespan, Cost and resource utilization.

1. Introduction

Maintaining rapid application development is an important aspect in the Information Technology sector and the reduction of time and effort put into the software deployment must be as minimal as possible. This calls for the usage of Cloud Computing. It is an upcoming trend that is widely used for purposes such as storage, sharing of memory, computational capacity sharing and hardware resource sharing over a network such as the internet. It is a concept that provides resources to both individuals as well as organizations, as a service that can be used at any given time or place of the user's demand and convenience. This results in the saving of time and cost for the users as they do not necessarily need to possess the resources they require, and can utilize the service at their will.

The major advantages of cloud computing are that it tackles important and necessary aspects like scalability, reliability, energy consumption, load balancing, time efficiency and cost efficiency. Of these tasks, allocation of resources is an important job to be performed by the network. This cannot be manually performed when a large number of virtual machines exist in the network and is therefore done by the machine layer using a prefixed optimized algorithm.

1.1 Cloud models

There are three types of models in cloud computing which are given as follows:

Public cloud model: The public cloud model is defined as a cloud computing infrastructure that is taken care of by a third-party service providing organization. This is available for both, individual users as well as software companies/organizations as a service over the internet. The major advantage of this model is that it is very large in scale. The users in this model share the same infrastructure pool with

limited configurations and security protection, as given by the service provider.

Private cloud model: The private cloud model is defined as a cloud computing infrastructure that is exclusive for each project or software being developed by a given company. This requires a permission policy to host applications in the cloud to enforce security and control in the system. Apart from being generated for every specific project, the cloud service can also be provided by an outside party or supplier.

Hybrid cloud model: The hybrid cloud model is defined as a cloud computing infrastructure that is a combination of the advantageous factors of both the public as well as private cloud models. This is accomplished using separate algorithms that are used to toggle between the two infrastructures.

1.2 Cloud computing services

Infrastructure as a service (IaaS) allows users to access the given network remotely to utilize its storage or computational units. It does so, on an on-demand basis for whenever the user requires the service. E.g.: Amazon Web Service, Microsoft Azure.

Platform as a service (PaaS) allows users to create web applications in a fast and simple way, with permissions, to provide a substitute for buying and maintaining the software and infrastructure to sustain the system. E.g.: Google App Engine.

Software as a service (SaaS) allows users to obtain a license for an application, to any user, either as a service on-demand or through subscription through the Internet. In simpler terms, rather than buying the required software, it can be rented for use in a pay-as-you-go manner. E.g.: Salesforce, Cisco WebEx.

1.3 Cloud computing tools

The cloud services over a network are used as efficient business solutions based on the organizational requirements. There are various cloud computing tools available like, Eucalyptus, Open Nebula, Nimbus, Openstack, etc. where all have different strategies for deployment.

Load balancing in cloud computing is defined as the process of distributing the workload and computing resources in a networked cloud computing environment. It allows an organization to manage applications or workload demands as per tasks, by allocating resources among the different computers on the networks or through servers.

1.4 Task scheduling

This is a process that occurs while using a constrained task based on the operation to be performed by the virtual machines. The data from the Request Manager or Server and Resource are collected by the scheduler and then computed to make a decision that allocates each task to its respective virtual machine.

In this proposed method we utilize makespan and cost among VMs as performance metrics to optimize task and resource, using an Oppositional Lion optimization algorithm (OLOA) algorithm based on the proposed models in cloud. The proposed scheduling hybridizes two algorithms namely Lion optimization algorithm and oppositional based learning (OBL). This new hybrid algorithm optimizes the task and resources more efficiently when compared with PSO, GSA and OGWO approaches. The comparative results obtained experimentally justified the efficiency of the proposed approach. The organization of this paper is as follows: Section 2 presents proposed scheduling using OLOA algorithm. Section 3 present the Result and discussion part. The conclusion part is given in section 4

2. Related work

In [1] Multi-criteria scheduling solution using algorithm Promethee algorithm is the proposed method for handling the challenges faced in a distributed network. Fault-tolerance, trust-awareness are some of the features of this method. Execution of large number of tasks upon a hybrid and flexible DCI is made possible. This method also performs massive multiplication of several other scheduling algorithms for comparison and demonstration purposes.

In [2] Minimizing the makespan and operating costs are the two major goals of this mathematical model proposed here. The mathematical model used is an NP-complete and a multi-objective genetic algorithm. This algorithm is based on a non-dominated sorting genetic algorithm. The computation and simulation results demonstrate that the proposed algorithm performs better than other algorithms.

In [3] Discrete Symbiotic Organism Search (DSOS) algorithm has been proposed for optimized task scheduling on the Cloud. It is a nouvelle metaheuristic technique that exhibits symbiotic relationships which exists in the ecosystem.

Simulation results depict that DSOS performs better than the Particle Swarm Optimization (PSO) algorithm. Convergence of DSOS is faster than PSO thus making it an OGWOl option for large-scale scheduling issues.

In [4] Two heuristics known as LSufferage and the TPB are proposed in this paper. ListSufferage is based on Sufferage. Tenacious Penalty Based heuristic increases quality. A mathematical model is depicted based on the Linear Programming method of Column Pricing. This method is termed as Column Pricing with Restarts (CPR). It is often considered as a hybrid and heuristic mathematical programming that solves issues in optimal time.

In [5] Security and Cost Aware Scheduling (SCAS) algorithm is proposed in this paper. This method handles heterogeneous tasks. It is based on Particle Swarm Optimization (PSO) algorithm. This is used to reduce the execution cost. Simulations with CloudSIM depict the efficiency of the proposed method.

In [6] on the basis of auction mechanism, an adaptive VM resource scheduling algorithm is proposed in this paper. The cloud user requests are queued in the competition deadline and the appropriate VMs are allocated according to the minimum costs of the service providers. Average payments and competitive payments are taken into consideration for the final payment. Experimentations reveal that this method increases the QoS of the Cloud.

In [7] BAT algorithm proposed in this paper is used to solve the multi-objective workflow scheduling problem in the Cloud in order to optimize execution time and reliability. Comparative simulations were made with the Basic Randomized Evolutionary Algorithm (BREA) and it was observed from the experimental results that BAT algorithm performs better than the other algorithm.

In [8] GA-ETI algorithm proposed in this paper permits easy adaptation to the various types of scientific workflows. The proposed system acts as an interface between the cloud user and cloud provider. Tasks such as receiving, analysing and distributing the tasks in the Cloud are carried out by this algorithm. Upon testing with five benchmarks of varied computing and data transfer demands was performed. The simulation results prove that the proposed algorithm decreases the makespan considerably on comparison with other existing scheduling algorithms.

In [9] this paper solves the optimization problem using makespan and cost, taking them as important constraints. Two algorithms known as the, cuckoo search algorithm (CSA) and oppositional based learning (OBL) are merged to create a new hybrid algorithm called the oppositional cuckoo search algorithm (OCSA). The proposed OCSA algorithm shows noticeable improvement over the other task scheduling algorithms. It is simulated in the cloudsim programming environment and the simulation results show the effectiveness of the proposed work by minimizing cost and makespan parameters. The obtained results are greater in performance in comparison to other existing algorithms like particle swarm optimization (PSO), oppositional learning based grey wolf optimizer (OGWO) and genetic algorithm (GA).

In [10] A hybridization of cuckoo search and gravitational search algorithm (CGSA) is proposed to lower the cost and resource and energy usage during task allocation. The advantages of two of the stated algorithms are present in this proposed method whereas the disadvantages are eliminated. Simulations were carried out by comparing the presented method with other algorithms like GSA, CS, etc.

In [11] An hybridised method of two algorithms namely the cuckoo search (CS) and harmony search (HS) algorithm is performed to attain optimization. New objectives are represented in the multi-objective function of this technique: cost, energy consumption, memory usage, credit and penalty. The performance of the proposed method is simulated and compared with different algorithms. It is found to have achieved a minimum cost, memory usage, energy consumption, penalty and maximum credit.

In [12] Genetic Grey Wolf Optimization Algorithm (GGWO) in combination with the grey Wolf Optimizer (GWO) and Genetic Algorithm (GA) is the technique that is proposed in this paper. Experimental results show that the GGWO improves task scheduling much better than standard GWO and GA with reduced computation time, migration cost and energy consumption.

In [13] Opposition Learning-based Grey Wold Optimiser Algorithm is the proposed hybrid technique that is used to reduce the makespan and cost of tasks in the Cloud.

In [14] Lion Optimization Algorithm (LOA) is a population-based algorithm that is proposed in this paper. The inspiration for this algorithm is derived from the wild lion's lifestyle and cooperation techniques. In [15-16] described about the task scheduling quality of service measure. In [17-19] described about the multitenant clouds.

The above described survey does not provide near optimum result by considering the parameters time, cost and resource taken together. Each of the

above mentioned approaches focus separately on these parameters whereas the proposed method considers these parameters together and provides an optimal solution in terms of task scheduling. In this proposed method we had utilized time, cost and resource usage among VMs as performance metrics to optimize them, using an Oppositional Lion Optimization Algorithm (OLOA). The proposed scheduling hybridizes two algorithms namely lion optimization algorithm (LOA) and oppositional based learning (OBL).

3. Problem definition with solution framework

The ultimate goal of the proposed methodology is to perform task scheduling with a minimum makespan and operating cost in a networked cloud system. The scheduling is done in parallel, where all the tasks are processed simultaneously. This plays an important role in directing tasks within the cloud network. The scheduling process analyses, and finds how many resources would be needed to complete the task and then finds which task should be allocated to which computing component in the network system.

In other words, a larger task can be split into smaller sub tasks before being sent for parallel processing. The computation can be much greater and much more efficient if a computation is broken down into smaller jobs and are executed on more than one processor. Scheduling all the jobs on a given number of processors, in order to increase the total final gain without changing the precedence of constraints is the resultant aim of the optimal task scheduling algorithm. It is an extremely challenging task for the system and therefore the optimization approach based scheduling has been proposed to overcome the difficulty in the present scheduling approaches to minimize cost of operation and execution time.

Table 1. Notation used in OLOA algorithm

Parameter	Definitions
PM_i	Physical Machine i $1 \le i \le n$
VM_i	Virtual Machine i, $1 \le i \le I$
T_i	Task i $1 \le i \le m$
C_i	Cost
E_i	Execution Time
α, β, γ	Decision variables
T_m	No of tasks
CPU_i	No of CPU
CCP	Computational Capacity Processor
N	No of Task

In this, a number of autonomous, simultaneous jobs are constituted in each task. Every job must be run on a single VM. Consider that $PM = \{PM_1, PM_2, ..., PM_n\}$ is a set of cloud physical Machine. $VM_i = \{VM_1, VM_2, ..., VM_I\}$ is a set of virtual machines (VM) types and $T = \{T_1, T_2, ..., T_m\}$ is a set of task.

$$Objective function = \sum_{i=1}^{m} T_i \begin{pmatrix} \alpha \cdot cost + \\ \beta \cdot Executiontime + \\ \gamma . Resource\ Utilization \end{pmatrix} \quad (1)$$

Cost: The cost of a task can be computed by the number of virtual machine movements divided by the total number of virtual machines on a physical machine. The cost function is given by :

$$C_i = \frac{1}{PM} \sum_{i=1}^{m} \left(\frac{Number\ of\ movement\ in\ VM}{Total\ VMs} \right) \quad (2)$$

Where C_i denotes cost, PM denotes Physical machine.

Execution time: The execution time is proportional to the task length and processing capacity. This function is expressed in the following equation:

$$E_i = \frac{Task\ Size}{CCP} \quad (3)$$

Where E_i denotes Execution Time, *TS* represents Task Size, *CCP* denotes Computational Capacity Processor

Resource Utilization: The resource utilization is used to determine the amount of (Memory, CPU, I/O) used in the physical machine.

$$Average\ Utilization = \frac{\sum_{i=1}^{m} \begin{matrix} Excecution\ Time\ of \\ task\ in\ paricular\ (Resource) \end{matrix}}{Makespan * N} \quad (4)$$

Where N denotes No of task.

3.1 Proposed OLOA based scheduling approach

Scheduling the task on the basis of OLOA algorithm is the main goal of the proposed system. It

is an hybridization of the lion optimization algorithm and Opposition based learning algorithm.

Step 1: Solution encoding

$$S_i = \begin{Bmatrix} t_1 cpu_1 & t_3 cpu_2 \cdots\cdots & t_n cpu_n \\ t_4 cpu_5 & t_5 cpu_3 \cdots\cdots & t_n cpu_n \\ t_2 cpu_1 & t_2 cpu_4 \cdots\cdots & t_n cpu_n \end{Bmatrix} \quad (5)$$

Step 2: Opposition based learning solutions

The opposite solution op $(z_1^*, z_2^*, z_3^* \ldots\ldots, z_n^*)$.

$$Z_{ij} = x_i + y_i - z_i \quad (6)$$

$i \in 1,2,3 \ldots n.$

Step 3: Fitness calculation

The fitness function is utilized to assess every task in view of the cost, makespan and resource. In this paper, the minimization capacity is taken as the fitness. For minimization issues, the fitness evaluation is performed by assessing the best and most exceedingly awful wellness for all specialists solutions at a number of iteration.

$$FF_i = \sum_{i=1}^{m} T_i \, (\alpha \cdot cost + \beta \cdot Execution\ time + \gamma . \ Resource\ Utilization) \quad (7)$$

Step 4: Update based of Lion optimization algorithm

The calculated fitness is then updated based on lion search, which is shown below:

$$\vec{S}_{(I+1)} = \vec{S}(I) + (0.1\overrightarrow{r_2} - 0.05)\left(\vec{S}(I) - \overrightarrow{r_1}\vec{S}(I)\right) \quad (8)$$

Step 5: Stop Criteria when it is satisfied.

The decision process of the proposed OLOA algorithm in a step-by-step procedure and the overall architecture of the proposed task scheduling is shown in Fig. 1. The working goes from the user specifying the task which is then handed over to the task scheduler. The task scheduler schedules the task based on the fitness function of each task after fitness calculation. The resource manager monitors the virtual machine usage which is taken as an observation for the given system proving that the Hybrid OLOA algorithm effectively reduces the cost of operation and the execution time.

Proposed Algorithm of OLOA

Input:

Number of task, Number of Host machines, Number of virtual machine.

Output:

Minimize Cost, Makespan and Resource Utilization.

Steps: 1 Initialize

Set value of parameters Number of Lions, VMs, Iterations.

Randomly select the initial population and opposition based.

For every single solution "LION"

Select randomly % N as Nomad lions and remaining as Residents.

For every Residents

Randomly divide into prides(P)

For each prides in Residents %S are considered female lion and remaining are males.

End

End

End

Step: 2 for each Pride

Select some females for hunting

Generate a pray at the center of the hunter as

$PREY = \frac{\sum position\ of\ hunter}{number\ of\ hunter}$

For i=1:H(H is number of hunters)

Move i^{th} hunter toward prey according to its appropriate group

If new place of i^{th} hunter is better than its last position

Prey escapes from hunter

End

End

Remaining females move toward best selected positions of territory.

Each male roams in %R of territory. %M of females mates with one or more resident males and produces two new off-springs

$$offspring1 = \beta \times Female\ Lion + \sum \frac{(1-\beta)}{\sum_{i=1}^{NR} S_i} \times Male\ Lion \times S_i \quad (9)$$

$$offspring2 = (1-\beta) \times Female\ Lion + \sum \frac{\beta}{\sum_{i=1}^{NR} S_i} \times Male\ Lion \times S_i \quad (10)$$

Sort males as mature according to their fitness value and weakest male drive out from pride and become nomad.

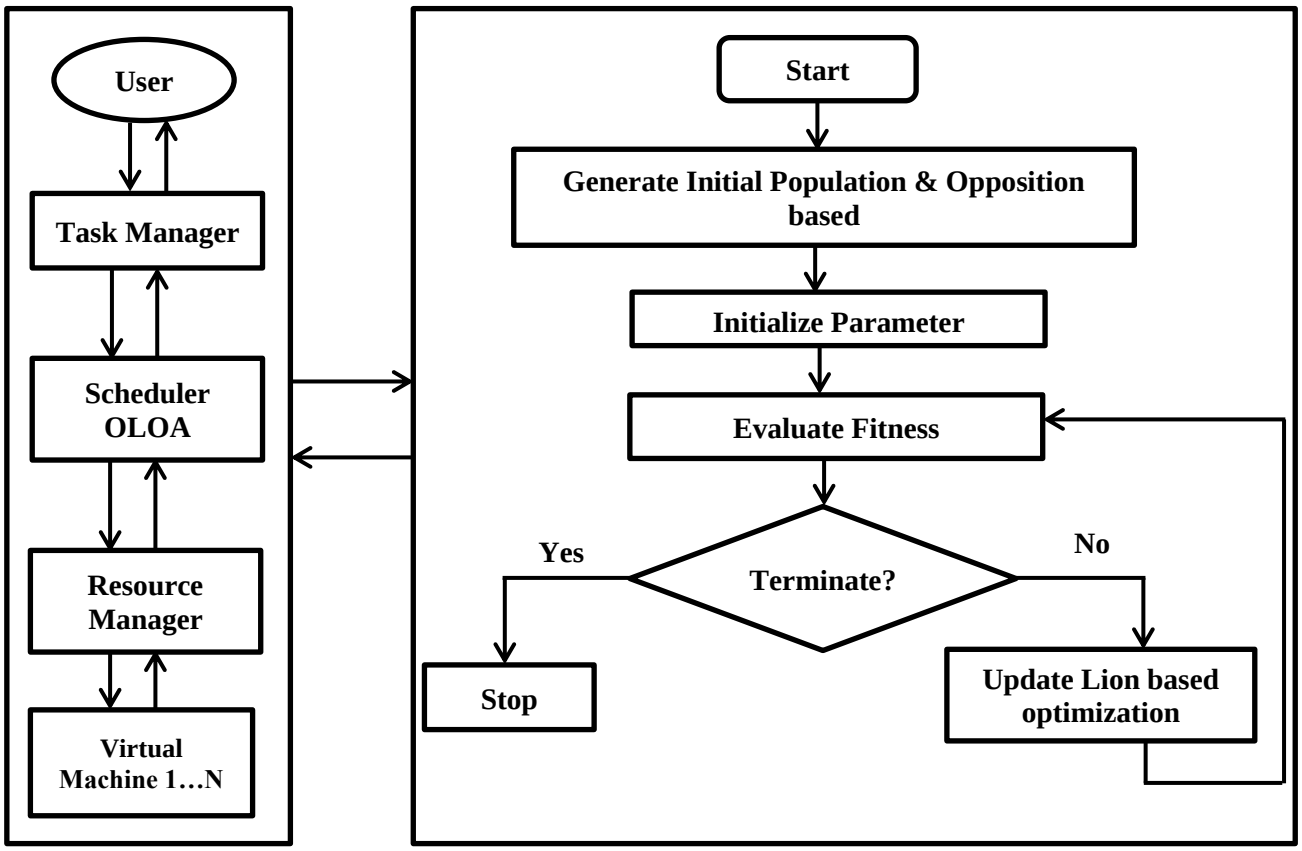

Figure. 1 Over all architecture diagram of task scheduling

3. For each Nomad lion

Both male and female move randomly in the search space Identify their new position as,

$$Lion = \begin{cases} Lion, & if \quad rand > pr \\ RAND, & otherwise \end{cases} \quad (11)$$

$$pr = 0.1 + min\left(0.5\frac{Nomad - BestNomad}{BestNomad}\right) \quad (12)$$

Where, rand is a random number between 0 and 1, pr is a probability, Nomad is the fitness value of the current nomad, and BestNomad is the best fitness value of the nomad lions. %M of females mate with only one male Nomad males attack prides.

4. For each pride,

Nomads are %I of the pride that is immigrated.

5. Do

On the basis of fitness value, both genders of the lion are sorted. Empty places are filled up with the optimal females. Those possessing the lowest fitness value will be eliminated according to the max permitted number from each gender.

6. If (t< Iterations)

Go to step 2

Return best solution.

4. Result and discussion

This section presents the output and inferences observed from the proposed hybrid OLOA scheduling algorithm. Our proposed OLOA is computationally compared with existing approaches – PSO, GA and OGWO. Both PSO and GA strive to provide and optimized solutions in scheduling tasks by repeatedly iterating the obtained candidate solutions. Similarly OGWO algorithm provides optimized solution to task scheduling problem by considering cost and time parameters. Since the proposed OLOA also focuses on providing a near optimal solution to task scheduling problem it has been compared with the aforesaid algorithms.

Here, the input tasks taken are of the range, 100 to 500. In our experiment, two cases with two different numbers of virtual machines were considered - 100 VMs and 200 VMs. This has been implemented using Java (jdk version 1.6) with the Cloudsim tool for simulation. A series of experiments were conducted on a desktop system with a 64-bit Windows 7 operating system with a 2 GHz dual core and 4 GB main memory. The results of proposed Hybrid OLOA are compared with PSO,

GA and with the result that is obtained in [13], where the author had used an oppositional learning based grey wolf optimizer (OGWO). The results obtained shows that our proposed technique produces better performance and cost when compared to other techniques.

4.1 Comparison of makespan

One of the main aspects of the proposed system is to reduce the execution time of the given task. The different algorithms, such as the OGWO, PSO and GA algorithms, and their performances have been compared with varied makespan values from the simulation. Fig. 2 compares the performances of the tasks and their execution time using 100 VMs. For the given tasks of 100, the values obtained are 95.2, 97.4, 98.3 and 100.7 for OLOA, OGWO, PSO and GA respectively. When the number of tasks is increased to 200, the values obtained are 170.5, 175.3, 180.5 and 188.4 for OLOA, OGWO PSO, and GA respectively. For 300 tasks, the values obtained are 200.2, 203.4, 205.3 and 210.6 for OLOA, OGWO, PSO, and GA, respectively. For 400 tasks, the values obtained are 250.1, 255.4, 260.3 and 280.6 for OLOA, OGWO, PSO, and GA respectively. For 500 tasks, the values obtained are 291.2, 294.4, 299 and 312.4 for OLOA, OGWO, PSO, and GA respectively.

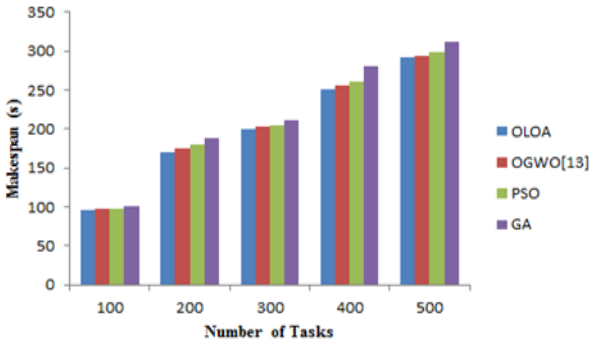

Figure. 2 Makespan of 100 VMs

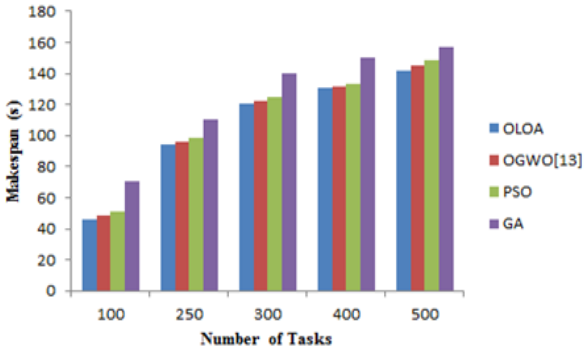

Figure. 3 Makespan of 200 VMs

From the Fig. 3 we infer that for 100 tasks, the corresponding makespan values of OLOA, OGWO PSO, and GA are 45.8, 48.7, 50.8 and 70.4 respectively. These results show the performance of OLOA is better considering double the number of VMs 200. When the task is increased to 200, the corresponding makespan values of OLOA, OGWO, PSO, and GA are 94.6, 95.8, 98.8 and 110.7 respectively. For 300 tasks, the corresponding makespan values of OLOA, OGWO, PSO and GA are 120.4, 122.1, 124.8 and 140 respectively. For 400 tasks, the corresponding makespan values of OLOA, OGWO, PSO, and GA are 130.5, 131.6, 133.5 and 150.4 respectively. For 500 tasks, the corresponding makespan values of OLOA, OGWO, PSO, and GA are 141.5, 144.7, 148.5 and 156.7 respectively. Thus the results obtained using OLOA makes it optimal for usage due to the significant difference (advantage) it provides when compared to rest. Also this difference (advantage) increases steadily as makespan increases to 200, 300, 400 and 500 indicating that OLOA yields better performance.

4.2 Comparison of cost

In Fig. 4 the performance metric is computed for analysing the maximum cost of the task per schedule. The maximum cost for 100, 200, 300, 400 and 500 tasks is determined using the chosen four algorithms. For the first computation done with 100VMs, the cost of 100 tasks is 65.2, 68.5, 69.5 and 90.7 for OLOA, OGWO, PSO and GA respectively, in which the OLOA is ahead of the rest. In the second computation, the cost of 200 tasks are 121.3, 125.8, 130.7and 150.8 for OLOA, OGWO, PSO, and GA respectively in which the OLOA is ahead of the rest. In third computation the cost of 300 tasks are 140.5, 145.8, 148.7 and 167.7 for OLOA, OGWO, PSO, and GA respectively, in which the OLOA is ahead of the rest. In fourth computation the cost of 400 tasks are 178.2, 180.7, 190.7 and 230.8 for OLOA, OGWO, PSO, and GA respectively, in which the OLOA is ahead of the rest. In fifth computation the cost of 500 tasks are 202.4,210,233 and 260 for OLOA, OGWO, PSO, and GA respectively, in which the OLOA is ahead of the rest.

Fig. 5 presents the last comparison for cost using 200 VMs that is carried using the same set of algorithms. It could be inferred from Figs. 4 and 5 that our OLOA approach shows an overall improvement with respect to cost and time. For 100 tasks the cost of OLOA, OGWO, PSO, and GA are 35.5, 38.9, 43.2 and 42.1 respectively. When jobs are 200 the cost of OLOA, OGWO, PSO, and GA

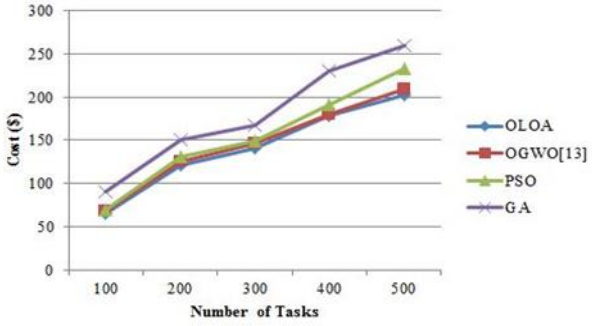

Figure. 4 Cost of 100VMs

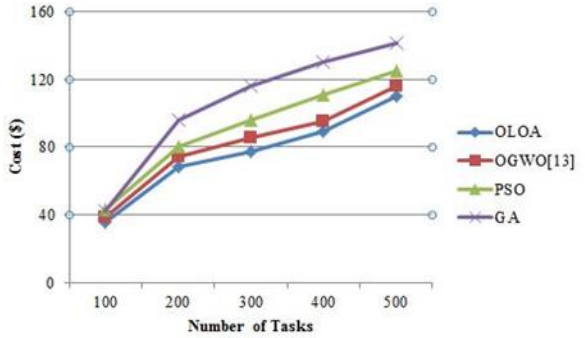

Figure. 5 Cost of 200VMs

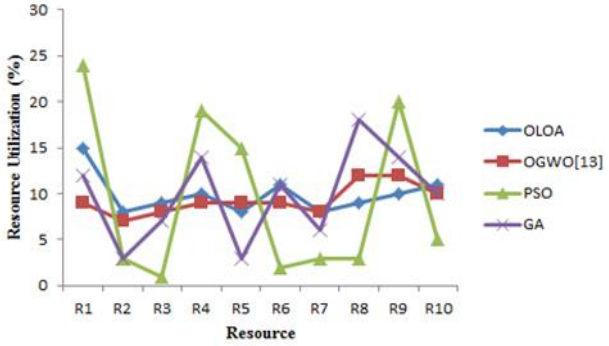

Figure. 6 Resource Utilization

are 68.4, 74.5, 80.5 and 95.7 respectively. For 300 tasks the cost of OLOA, OGWO, PSO, and GA are 77.7, 85.6, 95.8 and 115.8 respectively. For 400 tasks the cost of OLOA, OGWO, PSO, and GA are 89.5, 95.6, 110.7 and 130.3 respectively. For 500 tasks the cost of OLOA, OGWO, PSO, and GA are 110.3, 115.8, 124.8 and 141.4 respectively.

This clearly shows that the performance of OLOA increases even further when the numbers of tasks are increased to 200, 300,400 and 500. The cost of OLOA is much lesser when compared to PSO and OGWO. Also there is a drastic difference in cost when OLOA is compared to GA, thus proving the cost efficiency of OLOA.

4.3 Resource utilization

Fig. 6 shows the resource utilization during tasks that were executed in cloud environment. In this simulation, we considered ten resources and scheduled 100 times using four methods (OLOA, OGWO PSO, and GA) with varied deadlines. When the deadline is tight, the tasks tend to schedule several resources.

From the experiment results the proposed OLOA method is utilising the resources effectively when compared to other three methods.

5. Conclusion

In the proposed OGWO, the hybridization of two algorithms, Lion Optimization Algorithm and Opposition Learning Algorithm is done to propose the Oppositional Lion optimization algorithm (OLOA). The major objective of this scheduling technique is to optimally assign different machines, their tasks and to minimize the cost and makespan of the system as much as possible. This is a multi-objective optimization approach and is used to increase the scheduling performance. The experimental samples taken were based within a range of 100 to 500 tasks and 100 VMs and 200 VMs were considered. By performing hybridization of the Lion optimization algorithm and Oppositional Based Learning algorithm, a highly efficient solution for the scheduling mechanism is achieved. The results produced clearly show that the OLOA outperforms than OGWO, GA and PSO algorithms in terms of performance, cost and resource utilization. Therefore, in the future, more Quality of service (QoS) parameters can be integrated with our OLOA approach to extend its support for real time operations on a cloud network.

References

[1] M. Moca, C.Litan, G. Silaghi, and G. Fedak, "Multi-criteria and satisfaction oriented scheduling for hybrid distributed computing infrastructures", *Future Generation Computer Systems*, Vol.55, pp.428-443, 2016.

[2] H. Jiang, J. Yi, S Chen, and X Zhu, "A multi-objective algorithm for task scheduling and resource allocation in cloud-based disassembly", *Journal of Manufacturing Systems*, Vol.41, pp.239-255, 2016.

[3] M. Abdullahi and M. A. Ngadi, "Symbiotic Organism Search optimization based task scheduling in cloud computing environment", *Future Generation Computer Systems*, Vol.56, pp.640-650, 2016.

[4] C. Gogos, C. Valouxis, P. Alefragis, G. Goulas, N. Voros, and E. Housos, "Scheduling independent tasks on heterogeneous processors using heuristics and Column Pricing", *Future*

Generation Computer Systems, Vol.60, pp.48-66, 2016.

[5] Z. Li, J. Ge ,H. Yang, L. Huang , H. Hu, and B. Luo, "A security and cost aware scheduling algorithm for heterogeneous tasks of scientific workflow in clouds", *Future Generation Computer Systems*, Vol.65, pp.140-152, 2016.

[6] W. Kong, Y. Lei, and J. Ma, "Virtual machine resource scheduling algorithm for cloud computing based on auction mechanism", *Optik-International Journal for Light and Electron Optics*, Vol.127, No.2, pp.5099-5104, 2016.

[7] N. Kaur and S. Singh, "A Budget-constrained Time and Reliability Optimization BAT Algorithm for Scheduling Workflow Applications in Clouds", *Procedia Computer Science*, Vol.98, pp.199-204, 2016.

[8] I. Casas, J. Taheri, R.Ranjan, L. Wang, and A. Y.Zomaya, "GA-ETI: An enhanced genetic algorithm for the scheduling of scientific workflows in cloud environments", *Journal of computational science*, Vol.26, pp.318-331, 2016.

[9] P. Krishnadoss and P. Jacob, "OCSA: Task Scheduling Algorithm in Cloud Computing Environment", *International Journal of Intelligent Engineering and Systems*, Vol.11, No.3, pp.271-279, 2018.

[10] K. Pradeep and T. P. Jacob, "CGSA scheduler: A multi-objective-based hybrid approach for task scheduling in cloud environment", *Information Security Journal: A Global Perspe*c, Vol.27, No.2, pp.77-91, 2018.

[11] K. Pradeep and T. P. Jacob, "A Hybrid Approach for Task Scheduling Using the Cuckoo and Harmony Search in Cloud Computing Environment", *Wireless Personal Communications*, Vol.101, No.4, pp 2287–2311, 2018.

[12] N. Gobalakrishnan and C. Arun, "A New Multi-Objective Optimal Programming Model for Task Scheduling using Genetic Gray Wolf Optimization in Cloud Computing", *The Computer Journal*, Vol.61, No.10, pp. 1523-1536, 2018.

[13] G. Natesan and A. Chokkalingam, "Opposition Learning-Based Grey Wolf Optimizer Algorithm for Parallel Machine Scheduling in Cloud Environment", *International Journal of Intelligent Engineering and Systems*, Vol.10, No.1, pp.186-195, 2017.

[14] M. Yazdani and F. Jolai, "Lion optimization algorithm (LOA): a nature-inspired metaheuristic algorithm", *Journal of Computational Design and Engineering*, Vol.3, No.1, pp.24-36, 2016.

[15] K. Pradeep and T. P. Jacob, "Comparative analysis of scheduling and load balancing algorithms in cloud environment", In: *Proc. of International Conf. on Control, Instrumentation, Communication and Computational Technologies*, pp. 526-531, 2016.

[16] N. Gobalakrishnan and C. Arun, "SIS: A scheme for dynamic independent task scheduling in a cloud environment", In: *Proc. of International Conf. on Control, Instrumentation, Communication and Computational Technologies*, pp. 272-277, 2016.

[17] P. Rimal and M. Maier, "Workflow Scheduling in Multi-Tenant Cloud Computing Environments", *IEEE Transactions on Parallel and Distributed Systems*, Vol.28, No.1, pp.290-304, 2017.

[18] D. Zou, H. Liu, L. Gao, and S. Li, "An improved differential evolution algorithm for the task assignment problem", *Engineering Applications of Artificial Intelligence*, Vol.4, pp.616-624, 2011.

[19] G. Natesan and A. Chokkalingam, "Task scheduling in heterogeneous cloud environment using mean grey wolf optimization algorithm", *ICT Express*, 2018.