Multi-Objective Optimization for Dynamic Resource Provisioning in a Multi-Cloud Environment using Lion Optimization Algorithm

T Chaitra
Dept. of ISE
PESIT-Bangalore South Campus
Bangalore, India
chaitrad38@gmail.com

Shivani Agrawal
Dept. of ISE
PESIT-Bangalore South Campus
Bangalore, India
agrashiva@gmail.com

Jeny Jijo
Dept. of CSE
PES University
Bangalore, India
jenyjijo@pes.edu

Arti Arya
Dept. of CSE
PES University
Bangalore, India
artiarya@pes.edu

Abstract—**Cloud computing offers surfeit of services like storage, computing power, run-time environment, network etc. that everyone is accustomed to use it in day-to-day lives. In cloud computing, resources need to be dynamically provisioned on a metered basis. Quality of Services(QoS) is promised like performance, scalability, efficiency, fault tolerance, availability, reliability, throughput, and so on. Several meta-heuristic nature-inspired optimization algorithms like Particle Swarm Optimization(PSO), Ant Colony Optimization(ACO) etc. deployed to meet Service Level Agreement parameters like minimum downtime and low latency, but still have challenges in dynamically allocating resources. To overcome the above stated challenges , a new dynamic resource provisioning technique in a multi-cloud environment is proposed that uses Lion Optimization Algorithm(LOA) wherein characteristics of nomad and pride lion groups are taken into account. The multi-cloud environment provides the organization or customer to choose a provider that meets the specific requirements. As compared to PSO, this approach achieved better results while optimizing multiple objectives like completion time, average response time, makespan, cost, and average resource utilization. This study proves that the completion time and cost for LOA has outperformed when compared to PSO for a given number of tasks. The makespan and average response time for LOA improves slowly with more number of tasks as compared to PSO.**

Index Terms—**Cloud Computing, Resource Provisioning, Nature-inspired Algorithm, Lion Optimization Algorithm, CloudSim, Multi-cloud environment, multi-objective optimization.**

I. Introduction

Cloud computing has been dominant over all the preceding technologies like grid computing and other parallel and distributed computing. Cloud uses the concept of virtualization that is applied to the execution environment (application, programming language, operating system, and hardware) or compute level, storage level, and network level. It provides service like Software-as-a-service(SaaS), Platform-as-a-service(PaaS), and Infrastructure-as-a-service(IaaS). Companies deploy their private cloud which is used within their organization or public cloud to provide services [1]. There are community clouds and hybrid clouds for this purpose. These are different cloud deployment models.

As cloud computing is an on-demand service provider and when comes the management of these services, resource allocation is a major task that requires optimal solutions. The traditional allocation method is Static Allocation where service providers assign fixed resources to the cloud user which has a disadvantage of over-utilization or under-utilization and is not a cost-effective method. As the demand for resources and speed of processing increased, a new method is proposed called Dynamic Allocation where resources are provided based on the cloud consumer's request at any point in time [1]. This allocation method is used in different models that focuses on different Resource Allocation Strategies (RAS)[10] like Execution Time, Policy, VM, Utility, Service Level Agreement (SLA), etc. As demand increases, a single cloud provider cannot meet the exact needs of the cloud user. Thus two or more cloud providers can give flexibility to the user. This is called a Multi-cloud environment. In [8], authors describe that using a dynamic resource allocation in multi-cloud environment gives added advantages of faster resource availability, less resource wastage, etc. Overall execution time is improved when applications with independent tasks are deployed in multi-cloud environment. It provides access to a more feasible priced data centers, fault tolerance, high availability, and lesser dependency on one cloud. It speeds up the process by distributing the workload among the data centers . Care must be taken while migrating the VMs which is often based on the capacity. This saves energy consumption when VMs are consolidated. Scheduling is one of an integral part of resource allocation where tasks are mapped to appropriate resources. One of the traditional methods of scheduling is User level scheduling like First-Come-First-Serve (FCFS), priority based, non-preemptive scheduling , etc and scheduling is also an NP-Hard issue because of the huge solution space.In these methods, there is no room for improvisation. In [3] it is shown that traditional methods cannot obtain optimal solutions within polynomial time. In multi-cloud environment, resource provisioning takes more time to reach an optimal solution and belongs to the NP-hard problem because of huge solution space.

Various Meta-heuristic methods have proved that optimal solutions within a stipulated period for NP-hard problems can be achieved. There are several nature-algorithms like Ant Colony optimization, Genetic Algorithm, Particle Swarm Optimization, League Championship Algorithm, BAT algorithm, Wolf Optimization algorithm, Grasshopper optimization algorithm, etc. that are used in cloud computing for different purposes which focus on optimization. Each of the objective functions is measured in different units and are often contradictory to each other. For instance, when one tries to minimize time, the cost is not minimized and vice versa. In this paper, multi-objective optimization is done on five objectives namely cost, Completion Time(CT), Average Response Time(ART), Makespan and Average Resource Utilization(ARU) using a weighted sum approach.
The rest of the paper is organized as follows: Section II highlights the literature survey in the related field of nature inspired Lion Optimization algorithm (LOA) and application of such algorithms in the domain of multi-cloud environment. Section III gives a brief explanation of the LOA and Cloud Simulation. Section IV briefs about the proposed approach for optimizing multiple objectives using LOA in a multi-cloud environment. Section V discusses the implementation and results of the experiments conducted, followed by Conclusion in Section VI.

II. Related Work

There are several kinds of researches on optimizing the allocation of resources in cloud environment. In [1], Ab.Samah, et al.,2016, the authors provided a survey of models and solutions for the Resource Allocation problems. This paper also explains two major processes of Resource Allocation, namely, Static Allocation and Dynamic Allocation. It addresses several emerging challenges like the migration of VM, control mechanism over resources, energy efficiency, scheduling of parallel jobs, reduction of cost, and maximizing revenue, maintaining high availability and elasticity. To overcome these challenges, the focus is made on meta-heuristic algorithms rather than existing traditional models.

In [2], Janmenjoy et al., 2018, gives a detailed survey on optimization using nature-inspired algorithms in the areas of cloud computing. The main focus of this survey is on task scheduling, optimization of cost, load balancing etc. ACO and PSO were quite prominent in giving the optimal solutions. But few research challenges like mathematical and convergence analysis to get optimal condition, a trade-off between exploitation and exploration, accurately tuning the parameters of algorithms etc. are yet to be researched more in-depth.

Mala and Sarnjeet [3], provides a comparative study on these nature-inspired algorithms like PSO, Genetic Algorithm (GA), Championship Algorithm, ACO and others. It also discusses different objectives considered for optimization like the reduction of makespan, execution cost, time taken, etc. It is proved that meta-heuristic techniques achieve near-optimal solutions within a suitable time. The challenge is to reduce energy consumption of data centers without degrading performance and Service Level Agreement (SLA) violation. Prasad and Satyananda [4], focus on the issue of virtual machine placement, reducing resource wastage, energy consumption and communication cost by using ACO. This algorithm gave better performance in minimizing energy consumption and resource utilization when compared to other algorithms.

Feng Wang et al. [5], discussed about the effective results of using PSO for resource allocation. Here they have considered pareto-domination mechanism for multi-objective optimal solutions. The results are compared with their algorithms for different ranges of sub-tasks which are classified into small-scaled, middle-scaled and large-scaled instances. Sreelakshmi and SindhuIn [14], discusses and proves the efficiency of dynamic load balancing using multi-objective PSO where the metrics like makespan time and cost of communication is reduced thereby completing the task with the stipulated time. This was improvised by a hybrid firefly and Improved PSO (IPSO) where the problem of initial conditions and initial population did not give good results. Due to the selection of initial population using firefly algorithm the response time decreased when compared to IPSO. Then with the introduction of a hybrid algorithm that combined SA (Simulated Annealing) for selecting the global best position in multi-tier applications along with PSO gave better execution time when compared to IPSO alone [15], [17]. The author, D. Ardagna, in [16] brings the main disadvantages of a single cloud that are unpredictable performance, static VM service migration, poor SLA agreements, network management, and data center power management. To overcome these challenges the multi-cloud environment was adopted which also ensures minimum cost configuration satisfying the QoS constraints. In [8], the authors Panda, et al ., and in [18] the authors, I. Gupta, et al., explains about multi-cloud systems that are implemented in order to distribute tasks among clouds to manage peak-time client demands. For heterogeneous multi-cloud systems, three task scheduling algorithms are presented. The results show that these algorithms performed well when compared with the existing multi-cloud task scheduling algorithms in terms of makespan and average utilization. A two-phase workflow scheduling algorithm with a new priority scheme is also studied that is capable of scheduling large size workflows in a heterogeneous multi-cloud environment.

III. Preliminaries

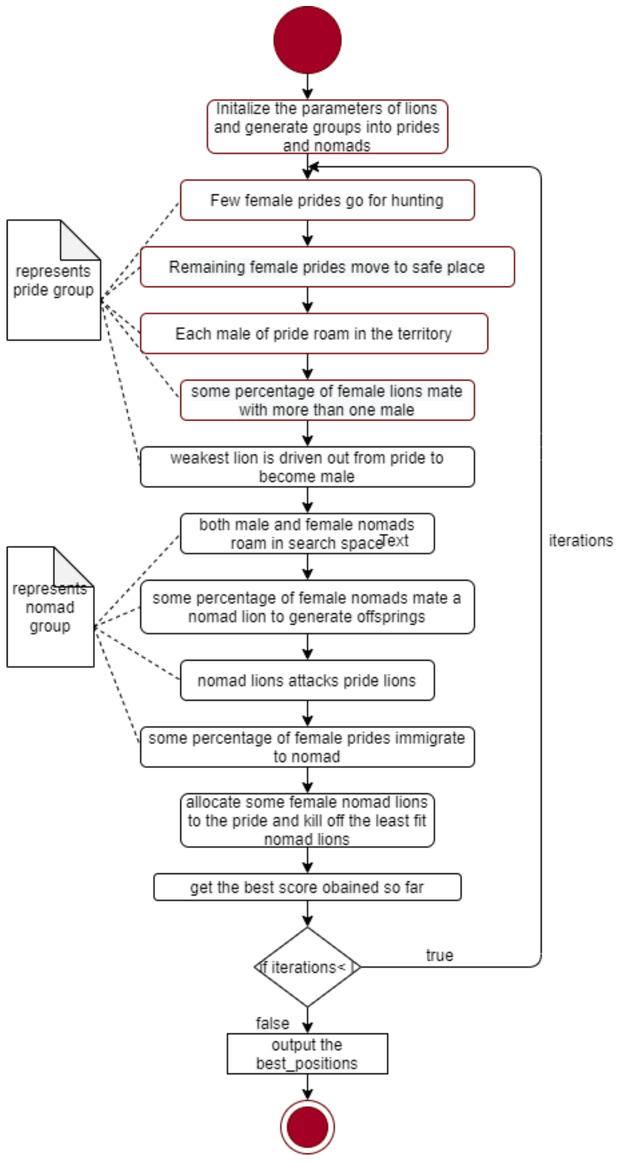
A. Lion Optimization Algorithm (LOA)

LOA is a computational meta-heuristic search based optimization algorithm inspired by the organizational behavior of lions. Lions are divided into pride and nomads based on how they stay. Pride lions stay in groups and nomads move in pairs or individually. The parameters used for this study are mentioned in TABLE I. The steps of LOA algorithm are as follows:

- **Generate Population:**
  Lion Optimization Algorithm is a population based

TABLE I
LOA PARAMETER VALUES

LOA Parameter	Value
npop	50
prideNo	4
percentNomad	0.2
roamingPercent	0.2
mutateProb	0.2
sexRate	0.8
mateProb	0.3
migrateRate	0.4



Initalize the parameters of lions and generate groups into prides and nomads
Few female prides go for hunting
Remaining female prides move to safe place
Each male of pride roam in the territory
some percentage of female lions mate with more than one male
weakest lion is driven out from pride to become male
both male and female nomads roam in search space
Text
some percentage of female nomads mate a nomad lion to generate offsprings
nomad lions attacks pride lions
some percentage of female prides immigrate to nomad
allocate some female nomad lions to the pride and kill off the least fit nomad lions
get the best score obained so far
if iterations<
true
false
output the best_positions
iterations
represents pride group
represents nomad group

Fig. 1. Lion Optimization Algorithm Flowchart.

method where initially the population is generated randomly in the solution space and every lion is considered as a solution. Based on the parameters specified in the table, the population is divided into pride and nomad group. The positions for the lions are randomly generated and these positions are the current best positions of lions. Based on these positions a territory is formed. In the following equations prey is represented as P, Hunter as H, Female lions as F_Lion. Male lions as M_Lion, Offspring as Cub, rand specifies random function.

- **Hunting:**
  In this step, some population of females from the pride group are selected for hunting. The selected female lions based on their fitness are divided into three groups namely left, right and centre wing. Fitness is defined by the position of each lion. Highest fitness group is considered to be centre and other two groups are left and right wings respectively [6]. Then a prey P is generated and initialized with a position to average of hunter positions, H_Position.

$$P = \frac{\sum H\_Position}{No.\ of\ Hunters} \tag{1}$$

  During hunting [6], hunters from each group are selected randomly for the attack on the prey. The new position for each hunter varies with the group.
  If the Hunter H belongs to left or right wing, the new position of Hunter, $H^{'}$ is obtained as follows [6]:

$$H^{'} = \begin{cases} rand((2*P-H),P)\ , & \text{(2*}P\text{ - }H\text{)} < P \\ rand(P,(2*P-H))\ , & \text{(2*}P\text{ - }H\text{)} > P. \end{cases} \tag{2}$$

  If the Hunter belongs to group centre, the new position of the lion, $H^{'}$ is obtained as follows:

$$H^{'} = \begin{cases} rand(H,P)\ , & \text{H} < P \\ rand(P,H)\ , & \text{H} > P \end{cases} \tag{3}$$

  In this step if $H^{'}$ position is better than H, Prey P will escape and a new position $P^{'}$ is assigned by the following formula:

$$P^{'} = P + rand(0,1) * (\%ofimprovement) * (P-H) \tag{4}$$

- **Move to safe Place:**
  The best positions [6] obtained so far over each iteration are saved for the improvement of the solution is stored in $K_j(s)$ ie., success rate.Tournament size, T_j^{Size} helps in expanding the territory of pride which helps for further functionalities like roaming.The size varies with the success rate, $K_j(s)$ and is calculated as follows:

$$T_j^{Size} = max\left(2, ceil\left(\frac{K_j(s)}{2}\right)\right) \tag{5}$$

where j represents pride group.
Remaining females of pride move to one of the areas of territory. The new positions of these female lions, $F\_Lion^{'}$ are obtained by

$$F\_ Lion^{'} = F\_ Lion + 2D * rand(0,1)\{X1\} + U(-1,1) + \tan(\theta) * D * \{X2\} \quad where \quad \{X1\}.\{X2\} = 0,\ ||\{X2\}|| = 1 \tag{6}$$

here D is the distance between the F_lion's position and point chosen by tournament selection.
{X1} and {X2} are the vector positions of start point and selected new position of female lion.

- **Roaming**
  Male Lions from the pride group roam in its territory [6]. Some percent of territory is randomly selected and are visited by the male lions. While roaming, if the visited position is better than the current position of lion, visited position is considered as lion's best position.
  Similarly in nomad, lions move randomly in search space. If the visited position is better than the current position of the lion, visited position is taken to be the best position of the lion [6]. Nomad lion's, $L_{ij}^{'}$ new position is calculated as follows:

$$L_{ij}^{'} = \begin{cases} L_{ij}, & rand_j > pr_i \\ RAND_j, & otherwise \end{cases} \tag{7}$$

  where L_{ij} represents nomad's current lion, j is dimension of the search space, RAND is a vector generated randomly in the search space, pr_i represents probability of the lion.

- **Mating**
  Based on mate probability and mutate probability, some percentage of female pride lions mate one or more pride males which are randomly selected. For nomads, females mates only one of the randomly selected males [6].
  New cubs, Cub_j1 and Cub_j2 are calculated by selecting the lions for mating as follows:

$$Cub_j1 = \beta * F\_ Lion_j + \sum \left(\frac{(1-\beta)}{\sum_{i=1}^{N} S_i} \right) * M\_Lion_j^i * mate_i \tag{8}$$

$$Cub_j2 = \big(1-\beta\big) * F\_ Lion_j + \sum \left(\frac{\beta}{\sum_{i=1}^{N} S_i} \right) * M\_Lion_j^i * mate_i \tag{9}$$

  where $mate_i$ is either 1 or 0 depending on selected for mating or not, N is the number of male lions in each group, β represents a random number generated with mean and standard deviation 0.5 and 0.1 respectively.
  The cubs calculated are randomly selected as female and other as male respectively and these offspring are added to pride and nomad groups.
- **Defence**
  In Pride group [6], resident lions with highest position score or fitness fights with other males and the weakest male is made to leave the pride to become nomad. Thus the strongest male is considered to be as a good solution.
  In nomad group, nomad males attack pride group randomly. If the nomad lions are strong, then the weak pride male is thrown out from the group to become nomad and the nomad lion becomes a pride.

- **Migration**
  Based on the migration rate, some percent of females from pride group move to become nomads. To maintain the balance of females in pride and nomad group, some nomad females move to become pride.

The above functionalities are shown in the flowchart shown in Fig. 1. The output of the algorithm are the best positions of lions and are obtained after a series of iterations through the various behaviour of lions over generations. These best positions are used for allocation of resources in the cloud.

B. Cloud Simulation

As explained in [7], the simulation goes through the following steps:

- The CloudSim is initialized with the current time which initializes the main CloudInformationService objects.
- Creation of cloud and Datacenter(s).
- Cloud broker is defined to simulate the task scheduling as well as VM allocation.
- The VMs are defined and the broker uses them for their DataCenters for its assignment and execution process.
- Define Cloudlet(s) and these cloudlets list are submitted to the cloudbroker for more task scheduling on the active VMs for its execution.
- Initiates the simulation.
- Cloudlets are allocated to VMs based on some algorithm and resources are allocated accordingly.
- Stop the simulation and flush all the entities.
- Print the simulation results where cloudlet ID, DataCenter ID , VM ID, start time, finish time ,execution time etc of each tasks are displayed in the console.

C. Multi-Objective Optimisation

Below are the multiple objectives considered for optimization

- **Make-span:**
  This objective helps in determining the maximum com-

pletion time of task execution. The the aim is to reduce the make-span.

$$makespan = max(endTime - startTime)_{task_i} \quad (10)$$

- **Average Response Time(ART):**
This objective helps in determining the average responses of each task execution. Based on the number of tasks, a number of data centers and virtual machines the average response time varies. The aim is to reduce the average response time.

$$ART = \frac{\sum (endTime - startTime)_{task_i}}{n}$$
$$where\ n\ is\ the\ total\ no.\ of\ tasks \quad (11)$$

- **Cost:**
Another objective is to assign the cost of resources based on the virtual machine's specification per hour. The aim is to minimize the resource cost.

$$cost = \sum (C_i * T_i)_{resource_i}$$
$$where\ 'C_i'\ is\ the\ cost\ of\ resource_i$$
$$'T_i'\ is\ the\ total\ time\ taken\ by\ resource_i \quad (12)$$

- **Completion time(CT):**
This objective helps in determining the total execution of all the tasks. The time determines the time taken for completion of the process. The aim is to reduce the execution time.

$$CT = \sum (endTime - startTime)_{task_i} \quad (13)$$

- **Average resource utilization(ARU):**
This objective determines the utilization of resources. The cloud provider should minimize the resources utilization in order to reduce energy consumption and dynamically assign incoming tasks to the remaining resources. The aim is to minimize the resource utilization.

$$ARU = \frac{\sum (finishTime - startTime)_{task_i}}{makespan * m}$$
$$where\ m\ is\ the\ total\ no.\ of\ tasks \quad (14)$$

For the evaluation of multiple objectives considered above: The objectives are aggregated into a single objective by assigning a weightage to each objective based on its importance. This method is called the Weighted Sum method which can handle many objectives in a short period of time.

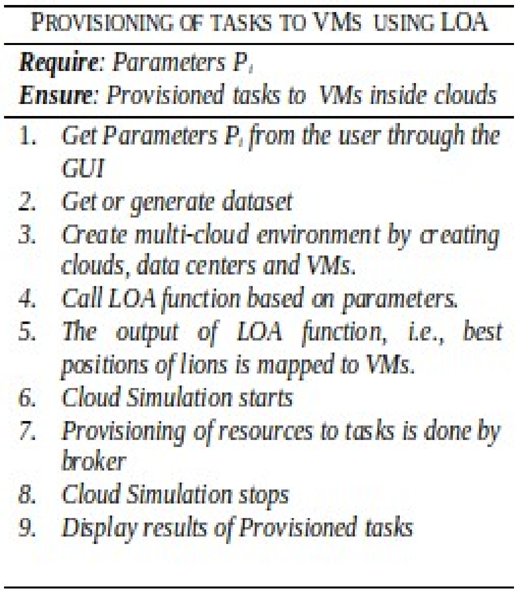
$$F(x) = \sum_{o=1}^{O} f_o(x).W_o \quad (15)$$

where:
O represents number of objectives
W represents the weight for each objective which ranges from[0-1] and $\sum_{o=1}^{O} W = 1$.
F(x) represents the weightage sum of x on multi-objective. $f_o(x)$ represents value of x for each objective o.

IV. Proposed Method

CloudSim simulation framework is used for simulating and modelling the proposed approach. The algorithm is coded in Java language to implement the cloud environment and run the application. CloudSim is mainly used for simulating cloud functionalities like cloud service broker, provisioning and allocation policies, datacenter, hosts, storage and virtual machines.

The pseudocode for the proposed method is as follows:



```
PROVISIONING OF TASKS TO VMs USING LOA
Require: Parameters P_i
Ensure: Provisioned tasks to VMs inside clouds
1. Get Parameters P_i from the user through the GUI
2. Get or generate dataset
3. Create multi-cloud environment by creating clouds, data centers and VMs.
4. Call LOA function based on parameters.
5. The output of LOA function, i.e., best positions of lions is mapped to VMs.
6. Cloud Simulation starts
7. Provisioning of resources to tasks is done by broker
8. Cloud Simulation stops
9. Display results of Provisioned tasks
```

Fig. 2. Provisioning using LOA Pseudocode

It uses LOA by applying the behavior of lions and give better results compared to other traditional meta-heuristic algorithm like PSO. Here we are applying LOA in multi-cloud environment by considering multiple objectives for optimizing Resource Allocation. The objectives are completion time, makespan, cost, average response time and average resource utilization. The proposed method provides optimized results for the above mentioned multiple objectives.
The parameters in TABLE II are used for the creation of cloud. These parameters are given by the cloud engineers who are the actual users of the system and use it to simulate a multi-cloud environment. Based on the requirement of cloud user, different clouds are created with different specifications in a multi-cloud environment.

Dataset for the proposed method is generated randomly based on the number of data centers and the number of tasks entered by the user. Each column denotes the data center like DC1, DC2, DC3 and so on. Similarly each row denotes the task T1, T2, T3 and so on. The dataset summarizes total time taken by each data center to complete each task. The cloudsim is trained to allocate the cloudlets based on the dataset.
Then LOA function is called for a specified number of iterations for better optimization towards the global maxima. The list of best positions of best lion which correspond the data

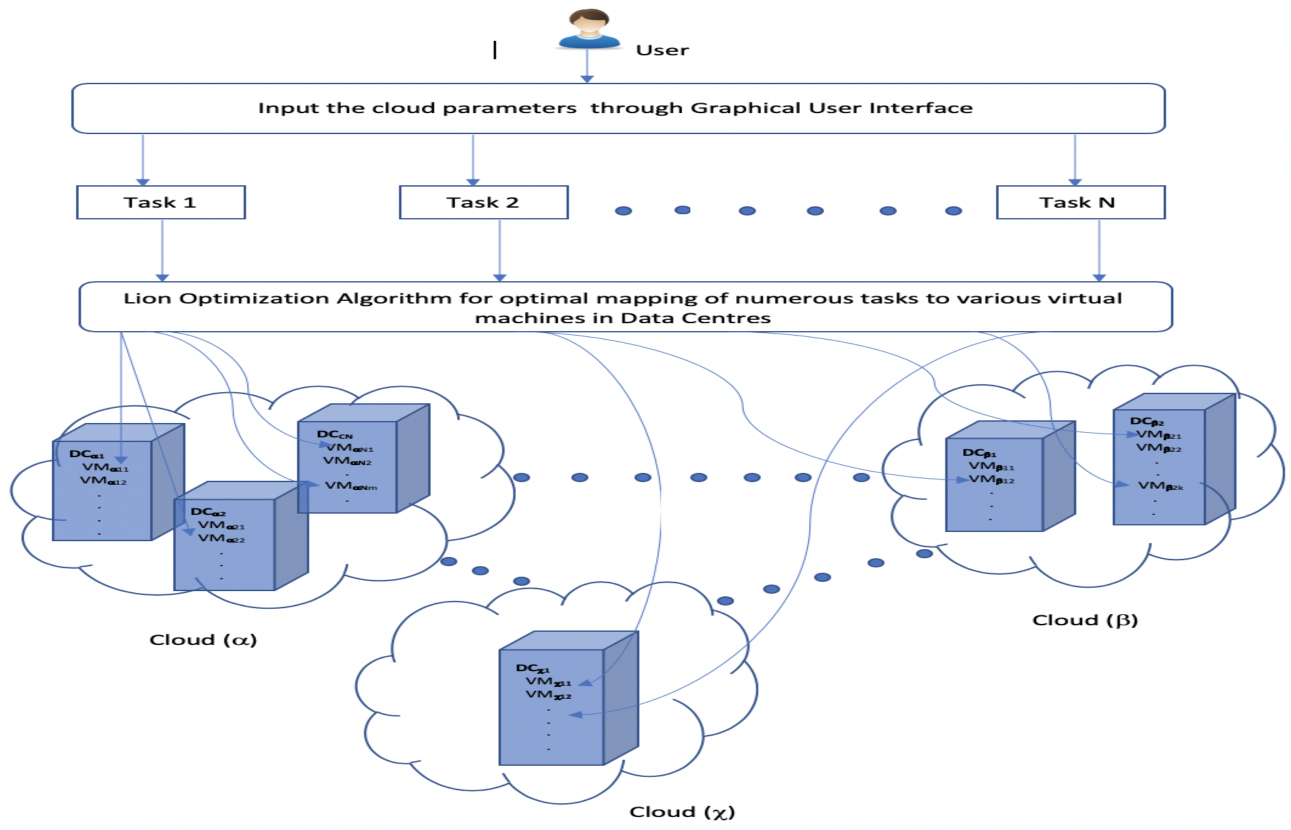

User
Input the cloud parameters through Graphical User Interface
Task 1
Task 2
Task N
Lion Optimization Algorithm for optimal mapping of numerous tasks to various virtual machines in Data Centres
Cloud (α)
Cloud (β)
Cloud (χ)

Fig. 3. The Proposed System Architecture

TABLE II
CLOUD PARAMETER VALUES

Cloud Parameter	Value
No. of tasks	10-100
No. of Data Centers	5-25
VM Size	5000-25000 (MB)
VM RAM	256-512 (MB)
VM MIPS	150-500
VM Bandwidth	500-5000 (MB/sec)
No. of CPUs	1-4
File Size	100-500 (MB)
Output File Size	200-500 (MB)

centers is obtained. The list is mapped to the actual data center IDs. The CreateBroker process is run and then it schedules the tasks to the VMs. Now when the simulation is started, the resources are best allocated based on the dataset. Finally, the simulation is stopped and a list of cloudlets along with its corresponding resource provisioned is printed with the time taken. The objectives are computed based on the equations (10) to (14).
Fig. 3 shows the proposed system for optimizing the multiple objectives using LOA meta-heuristic algorithm in a multi-cloud environment.

V. RESULTS AND DISCUSSION

The experiments are conducted on a machine having 64-bit operating system, windows 10 with 8GB RAM and 2.4 GHz processor. The program is run on Netbeans IDE having java, python and CloudSim jar files added where the multi-objectives of dynamic resource provisioning are optimized using LOA. The result of the proposed method, LOA is compared with a well known metaheuristic PSO and is shown in Fig.4, Fig.5, Fig.6, Fig.7 and Fig.8. LOA is executed for different number of iterations and was optimised around 225 iterations. The algorithm is run for different number of tasks : 25, 50 and 75. In the Fig.5 to Fig.8, red bars denote LOA and blue bars denote PSO. The x-axis represents number of tasks and y-axis represents different attributes in each figure.

Fig.4 shows bar graph for Completion Time Vs No. of Tasks compared between LOA and PSO. The y-axis represents Completion Time ranging between 0 to 15000 seconds. The objective is to minimize the completion time which is achieved by the proposed method. When compared to PSO, LOA takes much less time with increasing number of tasks also shown in Table III. Fig.5 shows Makespan Vs No. of Tasks compared between them. The y-axis represents Makespan ranging between 0 to 2200 seconds. The objective here is

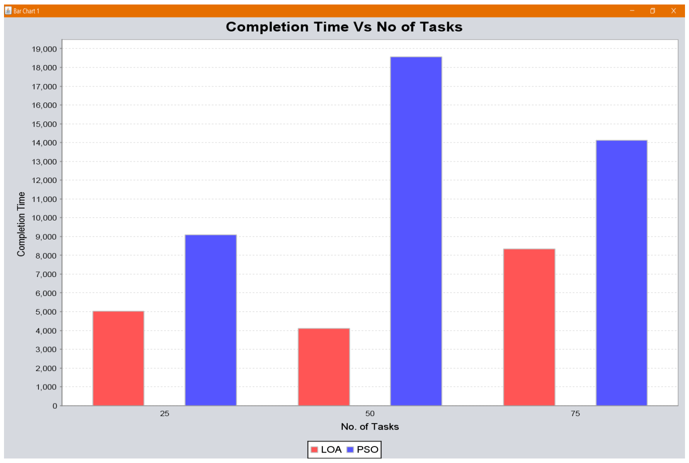


Fig. 4. Completion Time Vs No. of Tasks

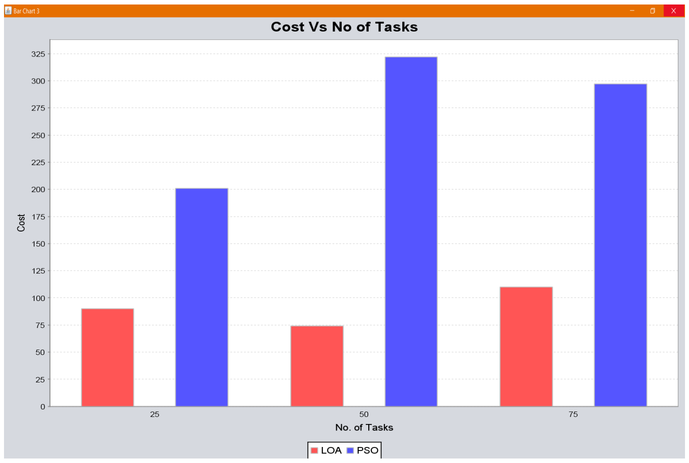


Fig. 6. Cost Vs No. of Tasks

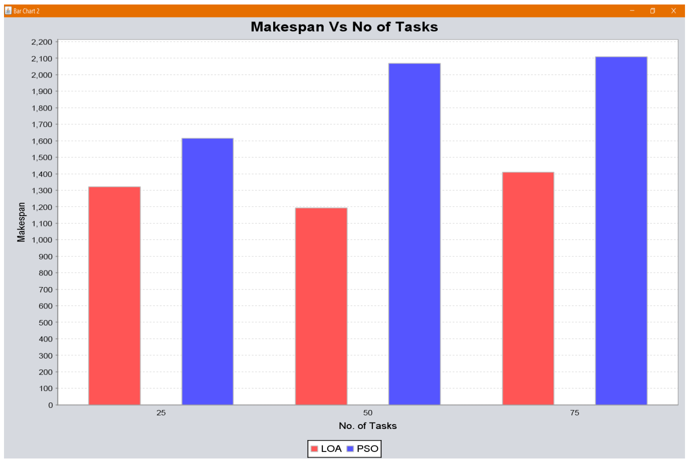


Fig. 5. Makespan Vs No. of Tasks

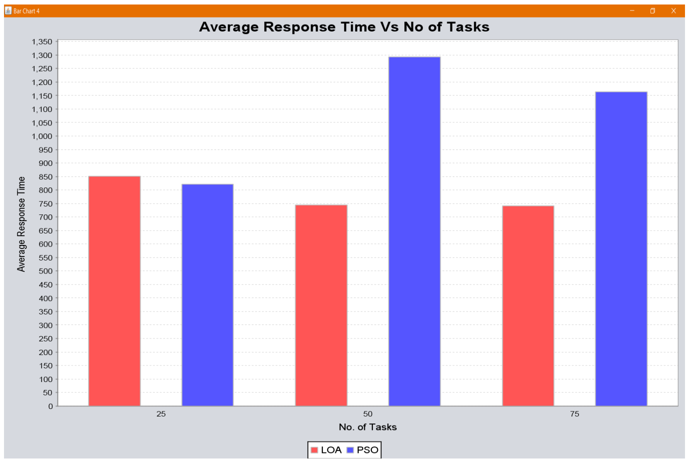


Fig. 7. Average Response Time Vs No. of Tasks

to minimize makespan which has been achieved and shown in TableIII. Initially, LOA takes approximately the same maximum response time as PSO. But with increasing number of tasks, LOA gave minimal increase in makespan which highlights the efficiency of applying LOA in place of PSO.

Fig.6 shows bar graph for Cost Vs No. of Tasks and the y-axis represents Cost ranging between 0 to Rs 325. The cost of VMs are considered as Rs 0.001, 0.015, 0.003 and so on per second. Clearly, LOA minimizes the cost involved in running the data centers as compared to PSO. Fig.7 shows bar graph for Average Response Time Vs No. of Tasks and the y-axis represents Average Response Time ranging between 0 to 1250 seconds. Here the objective is to minimize the Average Response Time which is achieved by the proposed method. For 25 tasks, PSO gave better result but as the number of tasks is increased, LOA minimizes the average response time as mentioned in TableIII.

Fig.8 shows bar graph for Average Resource Utilization Vs No. of Tasks compared between LOA and PSO. The y-axis represents Average Resource Utilization ranging between 0 to 5.25. The aim is to reduce the wastage of resources used in the form of VMs assigned. LOA uses very few resources even for more number of tasks.

TABLE III
PERFORMANCE COMPARISON OF LOA OVER PSO

Objectives	PSO	LOA	PC(%)
CT(ms)	18564	4117	77.82
Makespan(ms)	2070	1192	42.41
Cost(Rs)	323	74	77.089
ART(ms)	1294	745	42.42
ARU(%)	6.25	2.08	66.72

For the given parameters , No. of tasks = 50, No. of Data centers = 15, VM size = 15000 MB, VM Ram =512 MB, VM MIPS = 400, File Size = 300 MB, VM BW = 4000 MB/sec, Output File size = 400 MB, No. of CPUs = 2, the results for different objectives considered are compared in TABLE III. The Performance Comparison (PC) column represents the percentage by which LOA outperforms with respect to PSO.

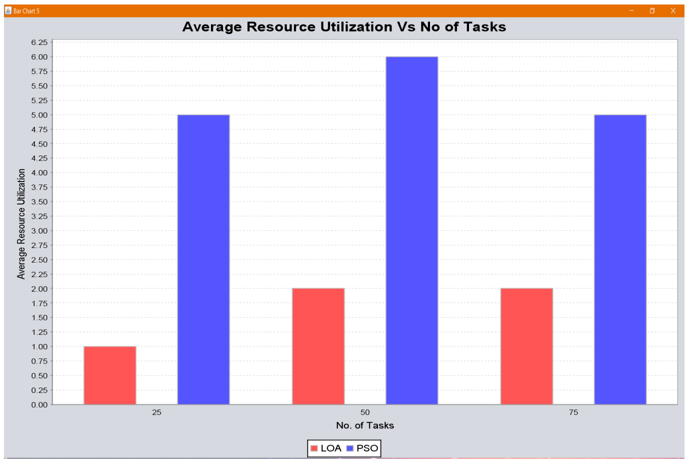


Fig. 8. Average Resource Utilization Vs No.of Tasks

It is calculated as follows:

$$PC\% \quad = \quad \frac{(Value_{LOA} - Value_{PSO})}{Value_{PSO}} \, * \, 100 \qquad (16)$$

Thus our proposed method, LOA performs better than PSO for all the objectives considered and gives the best result for CT followed by cost, ARU, ART and makespan. The implementation is available at Github [19].

VI. Conclusion and Future Scope

In multi-cloud environment, there were issues regarding performance in allocating resources dynamically. This paper addresses performance, efficiency and reliability issues of resource provisioning. The proposed method using LOA outperforms dynamic resource allocation for all considered objectives. The result of the proposed method shows better percentage of improvement as compared with the traditional PSO algorithm. The objective like the total execution time for resource allocation that made the system finish the job faster is met. The cost involved in running the infrastructure and processes is minimized. The minimized makespan tells that maximum response time is less among the pool of processes executed. The average response time is minimized from which we can infer that the average waiting time for individual task allocation is also minimal. The average resource utilization is reduced by more than 50% for 50 tasks which is encouraging and the energy consumed is saved to run the data centers. This is indicative of less usage of resources so that we can allocate further upcoming tasks on the remaining resources. This helps save energy, reduce cost involved and make more revenue from the unused infrastructure.

In the future, we have planned to consider the performance of the same parameters used in LOA and simulate them in the federated cloud environment . There will be a study also on how to incorporate Live VM migration as an effective mechanism to balance the load among the different Cloud Service Provider (CSP) using metaheuristic algorithms.

References

[1] Ab, Samah & Dogan, Meram & Alqahtani, Ebtdesam, A SURVEY ON RESOURCE ALLOCATION IN CLOUD COMPUTING", International Journal on Cloud Computing: Services and Architecture (IJCCSA). 6.10.5121/ijccsa.2016.6501.

[2] Janmenjoy Nayak, Bighnaraj Naik, AK Jena, Rabindra K Barik, and Himansu Das, Nature inspired optimizations in cloud computing: applications and challenges", In Cloud Computing for Optimization: Foundations, Applications, and Challenges, pages 1–26. Springer, 2018

[3] Mala Kalra, Sarbjeet Singh, A review of metaheuristic scheduling techniques in cloud computing", Egyptian Informatics Journal (2015) 16,275-295.

[4] Prasad Devarasetty, Ch. Satyananda Reddy," Multi objective Ant colony Optimization Algorithm for Resource Allocation in Cloud Computing", International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-8 Issue-2S2 December, 2018.

[5] M. Feng, X. Wang, Y. Zhang and J. Li, "Multi-objective particle swarm optimization for resource allocation in cloud computing," 2012 IEEE 2nd International Conference on Cloud Computing and Intelligence Systems, Hangzhou, 2012, pp. 1161-1165, doi: 10.1109/CCIS.2012.6664566.

[6] 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.

[7] N. Calheiros, Rajiv R, Anton B, Rajkumar B, CloudSim: a tool kit modeling simulation of cloud computing environments and evaluation of resource provisioning algorithms", Wiley Online Library, DOI: 10.1002/spe.995, 24 August 2010.

[8] Panda, S. K., & Jana, P. K. (2015). Efficient task scheduling algorithms for heterogeneous multi-cloud environment. The Journal of Supercomputing, 71(4), 1505–1533. doi:10.1007/s11227-014-1376-6

[9] S. Thamarai Selvi, Christian Vecchiola, Rajkumar Buyya, "Mastering Cloud Computing", Morgan Kaufmann, Edition ,2013.

[10] Anuradha VP, Sumathi D. A survey on resource allocation strategies in cloud computing. International Conference on Information Communication and Embedded Systems (ICICES); Chennai. 2014. p. 1–7.

[11] Kumar S, Kumar Sharma V, Kumari R (2014) Self-adaptive spider monkey optimization algorithm for engineering optimization problems. Int J Inf Commun Comput Technol II:96–107

[12] J. Vahidi and M. Rahmati, "Optimization of Resource Allocation in Cloud Computing by Grasshopper Optimization Algorithm," 2019 5th Conference on Knowledge Based Engineering and Innovation (KBEI), Tehran, Iran, 2019, pp. 839-844, doi: 10.1109/KBEI.2019.8735098.

[13] M. A. Tawfeek, A. El-Sisi, A. E. Keshk and F. A. Torkey, "Cloud task scheduling based on ant colony optimization," 2013 8th International Conference on Computer Engineering Systems (ICCES), Cairo, 2013, pp. 64-69.doi: 10.1109/ICCES.2013.6707172

[14] Sreelakshmi, S.Sindhu "Multi-Objective PSO Based Task Scheduling - A Load Balancing Approach in Cloud",1st International Conference on Innovations in Information and Communication Technology (ICIICT), IEEE, 2019.

[15] Mahya Mohammadi Golchi, Shideh Saraeian and Mehrnoosh Heydari,"A hybrid of firefly and improved particle swarm optimization algorithms for load balancing in cloud environments: Performance evaluation", Computer Networks, ELSEVIER, 2019.

[16] D. Ardagna, "Cloud and Multi-cloud Computing: Current Challenges and Future Applications," 2015 IEEE/ACM 7th International Workshop on Principles of Engineering Service-Oriented and Cloud Systems, Florence, 2015, pp. 1-2, doi: 10.1109/PESOS.2015.8.

[17] Marwah Hashim Eawna, Salma Hamdy Mohammed, El-Sayed M. El-Horbaty, "Hybrid Algorithm for Resource Provisioning of Multi-tier Cloud Computing",Procedia Computer Science, Volume 65,2015.

[18] I. Gupta, M. S. Kumar and P. K. Jana, "Compute-intensive workflow scheduling in multi-cloud environment," 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Jaipur, 2016, pp. 315-321, doi:10.1109/ICACCI.2016.7732066.

[19] https://github.com/ShivaniSarah/Metaheuristic-Dynamic-Resource-Provisioning-in-Multi-Cloud-Environment/blob/master/README.md