Efficient Resource Management using Ant Lion Optimisation Algorithm

MSc Research Project
MSc in Cloud Computing

Aditi Dilip Sulke
Student ID: 22138617

School of Computing
National College of Ireland

Supervisor: Prof. Punit Gupta

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Aditi Dilip Sulke
Student ID:	22138617
Programme:	MSc in Cloud Computing
Year:	Jan 2023-24
Module:	MSc Research Project
Supervisor:	Punit Gupta
Submission Due Date:	14/12/2023
Project Title:	Efficient Resource Management using Ant Lion Optimisation Algorithm
Word Count:	
Page Count:	23

Signature:	Aditi Dilip Sulke
Date:	14th December 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Efficient Resource Management using Ant Lion Optimisation Algorithm

Aditi Dilip Sulke
22138617

Abstract

This study investigates an in-depth comparison of meta-heuristic algorithms, Ant Lion Optimizer (ALO) and Ant Colony Optimisation (ACO) in the context of Execution time and VM allocation in Cloud Computing. It determines which algorithm produces superior results by focusing on execution time efficiency and differences in VM allocation. The research begins with a thorough examination of both algorithms, emphasizing their underlying principles and applications in the context of resource allocation, followed by a comparative analysis of the performance efficacy of these algorithms. The effects of these algorithms concerning task execution time which is one of the critical metrics in cloud computing is evaluated and its comparison sheds light on how both of these algorithms affect resource utilisation. This study offers useful insights for practitioners looking for optimal VM allocation strategies, emphasizing ALO's advantages over ACO in terms of execution time and adaptability. By the end of this study, ALO emerges to have less execution time and maximum resource utilization can be visualized. To improve overall system performance, the primary focus is on optimizing Virtual Machine (VM) allocation and minimizing execution time. As the research concludes, ALO emerges as an optimal solution with shorter execution times and better resource utilization, implying its potential for improving overall system performance.

1 Introduction

The rapid growth of cloud computing in recent years has changed the dynamics of IT infrastructure management. Cloud services are attracting businesses with the promise of on-demand resources, scalability, and cost-effectiveness. However, as cloud infrastructures grow in complexity and scale, efficient resource allocation has become a critical issue. The need to address the challenges posed by dynamic workloads, changing requirements, and the imperative to optimize resource utilization in cloud environments is driving this research. The role of optimization algorithms in ensuring efficient resource allocation is becoming increasingly important as the field evolves. ALO and ACO, inspired by the collective behavior of ants and ant lions, have shown promise in solving complex optimization problems. This research aims to evaluate their practical application in cloud computing, with a particular focus on VM allocation and execution time efficiency.

An essential aspect of cloud infrastructure management is the allocation of Virtual Machines (VMs), a task that directly influences resource utilization, system efficiency, and overall performance. As the scale and complexity of cloud environments continue to grow, the optimization of VM allocation becomes paramount for achieving optimal

resource utilization and responsiveness.
Ant Lion Optimizer (ALO) and Ant Colony Optimization (ACO) have emerged as powerful nature-inspired optimization algorithms, drawing inspiration from the foraging behaviors of ants and ant lions. These algorithms have proven to be effective in solving complex optimization problems, such as those found in cloud computing. This thesis uses a simulation tool to conduct a comparative analysis of ALO and ACO in the context of VM allocation within cloud computing environments. The primary focus is on evaluating the performance of these algorithms in terms of execution time efficiency and resulting VM allocation patterns. Understanding the advantages and disadvantages of each algorithm in this context is critical for making decisions in real-world cloud deployments.

Although there are many other service delivery models in this technology, the focus of this research is on the Infrastructure as a Service (IaaS) model. It is concerned with the server side of this technology for resource allocation. Task scheduling allows virtualized resources to be assigned to a specific task for a set period. It is possible to accomplish this by utilizing a task-scheduling algorithm that will be handled by a cloud resource broker. Task scheduling indicates that the next task will be completed in the shortest amount of time. CSP monitors the status of running virtual machines to find a better resource for an upcoming task. Following that, it performs a load-balancing operation to keep all VMs loaded.

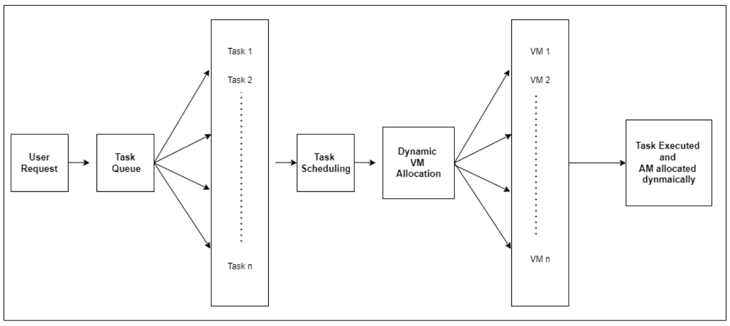

Figure 1: Task Scheduler Flow

In the above Figure.1, A user's request to run a task or application on the cloud platform is represented by a User Request. A queue that manages incoming user tasks. Tasks are prioritized based on their arrival time. The task scheduler is in charge of assigning tasks from the queue to available virtual machines (VMs) or containers. It uses scheduling algorithms to optimize task allocation based on factors such as resource needs, priority, and load balancing. The dynamic resource allocator continuously monitors VM resource utilization and makes dynamic decisions to allocate or deallocate resources based on demand and performance metrics. It entails scaling up or scaling down VMs, adjusting CPU and memory allocations, and provisioning additional VMs. The VM pool is a collection of available virtual machines. Dynamically added and removed virtual machines.

In basic terms, this research question explores the practical world, attempting to uncover the implementation complexities and tangible impact of optimisation algorithms in the context of cloud computing. The overall objective is to improve resource management, refine task scheduling methodologies, and boost cloud system efficiency. The Cloudsim framework is chosen for this implementation because of its ability to simulate cloud environments and facilitate a detailed analysis of optimization algorithm performance.

1.1 Research Background

In the field of information technology, cloud computing has become a paradigm-shifting phenomenon that is completely changing how resources are managed and delivered. The scalability and flexibility of cloud services are attracting more and more organizations, making resource allocation within cloud environments crucial. The management of virtual machines (VMs), which has an impact on resource utilization, system efficiency, and overall performance, is an essential aspect of this allocation process. Nature-inspired approaches have become popular in the optimization algorithms field for solving challenging cloud computing issues. This research compares two metaheuristic algorithms that are based on the collective behavior of ants and ant lions: Ant Lion Optimizer (ALO) and Ant Colony Optimization (ACO). These algorithms apply to the dynamic and complex nature of optimization problems because they have proven to be successful in solving them.

The research is being driven by the increasing size and complexity of cloud infrastructures. To achieve optimal resource utilization and responsiveness, VM allocation must be optimized. To make informed decisions in practical cloud deployments, it becomes essential to understand how ALO and ACO perform in terms of execution time efficiency and VM allocation. As cloud computing evolves, this study focuses on the Infrastructure as a Service (IaaS) model. Cloud Service Providers (CSPs) manage server-side infrastructure such as data centers and servers in IaaS. Algorithms for allocating virtualized resources to specific tasks are essential, and cloud resource brokers handle this scheduling. The focus on IaaS emphasizes the importance of effective resource allocation, task scheduling, and load balancing for optimal cloud service delivery.

1.2 Research Question

The focus of this research is on the efficient use of optimization algorithms within cloud environments, and how optimization algorithms can be used to improve resource allocation, such as Virtual Machines (VMs) and other computational resources. Examine the effect on resource utilization. It is critical to evaluate how optimization algorithms handle dynamic workloads and contribute to the scalability and adaptability of cloud environments to meet the changing needs of cloud-based applications. The research question dives into the practical implementation and impact of optimization algorithms in cloud computing environments, to improve resource management, task scheduling, and overall system efficiency.

1.3 Research Objective

This research compares the performance of two metaheuristic algorithms in cloud computing, Ant Lion Optimizer (ALO) and Ant Colony Optimization (ACO), in terms of execution time and virtual machine (VM) allocation. The following are examples of primary research objectives:

- Evaluating and comparing the efficiency of ALO and ACO in terms of execution time, to determine which algorithm performs better.

- Examining and comparing the effectiveness of ALO and ACO in virtual machine allocation, with a focus on understanding how each algorithm optimizes resource utilization.

- Exploring the efficiency of each algorithm to allocate resources as the number of tasks or workload intensity changes.

- Contributing to the study of optimization strategies by providing comparative analysis.

1.4 Outline

The Related works for this study is given in Section 2, followed by Methodology in Section 3. Section 4contains the Design Specifications related to the study. Section 5 has the content related to implementation of this study, followed by Evaluation in Section 6. Finally the research is concluded in Section 7

2 Related Work

This paper gives a review of the whole list of various algorithmic approaches. A number of algorithms have been mentioned that work on load balancing, and task scheduling. The author has proposed a two-level architecture for load balancing in which the first level is performed on the physical machine and the second on the VM Level. Based on this proposal there are two sets of task migration namely Intra VM task migration and Inter VM task migration. However, the load balancing activity involves steps for the identification of user task requirements, VM resource details, resource allocation, task scheduling, and migration. The author has also identified major problems rising to load unbalancing which include the dynamic nature of tasks, unpredictable traffic flow, and demanding resource requirements. The major performance metrics considered in this study are response time, performance, makespan, throughput, resource utilization, migration time, scalability, energy consumption, and carbon emission. The only drawback is that this paper is not able to give a specific best approach for load balancing algorithm which can be used to resolve the mentioned issue but is able to give a good review of the list of algorithms.Afzal and Kavitha (2019)

In this research paper, the author has claimed that Virtualisation is the backbone and most essential feature of cloud-based applications, which significantly affects the performance of scalable services that are provided on demand by clients if the migration process and allocation of virtual machine resources are handled inefficiently. This study tries to improve resource allocation in the IaaS paradigm; this idea is important because it deals with the balance of resources offered to clients and workload/user demands on servers. Cloud users access services by making requests, which are represented in the cloud environment by Virtual Machines (VMs). This researcher proposes an algorithm for load balancing which mainly is focused on the IaaS model. The author has implemented the proposed load balancing algorithm using simulation, and according to them task scheduling significantly adds to load balancing in a cloud system. Improving the Load Balancing process with Task Scheduling can lead to more efficient use of cloud resources. The goal of this work was to present an improved Load Balancing method. When compared to conventional Dynamic LBA, our technique minimizes Makespan and provides 78 percent more efficient resource use.Shafiq et al. (2021)

The research paper provides preliminary knowledge about various techniques of load balancing and also answers the question of why load balancing is needed. The author has mentioned briefly the challenges of load balancing and has discussed the same. Load balancing techniques are used to balance the load on virtual cloud computers, allowing each machine to operate equally based on its capacity. Load balancing allows work to be delivered evenly to each virtual machine, resulting in the optimum performance from each. Cloud service providers may control strain on virtual machines using load-balancing strategies. Cloud computing infrastructure is built on virtual computers. The author has mentioned challenges that can be used to narrow down the identification problem for load balancing can be listed as geographically distributed nodes, single point failure, virtual machine migration, heterogeneous nodes, load balancing scalability, and algorithm complexity. The techniques for VM migration are of great use as their classification table can be taken into consideration to identify the pros and cons of workload balance which can lead to Low task execution and reduced response time.Sriram (2022)

In this research paper, the author has proposed a 3-tier architecture consisting of Cloud Layer, Fog Layer, and Consumer Layer as a solution in response to the optimization of the load balancing feature. As the size of cloud data centers increases, there is a growth in the number of virtual machines. VMs placed on the physical machine (PM) serve application requests. The fast expansion of Internet services has resulted in a network resource imbalance. Some hosts consume a lot of bandwidth, which might cause network congestion. Network congestion has an impact on overall network performance which leads to the solution of optimising load balancing. In this research, one more aspect of live virtual migration was done because while considering the constant fluctuation of VM load, the physical host load may be high enough to impair service quality, or it may be too low to fully use resources. The VM migration was necessary to improve service quality and resource utilization rate. For simulation, Cloud Analyst Simulation tool was used. To efficiently balance the load of VMs in the fog, this study suggests live VM migration. However, processing time is increased since the live VM migration algorithm continually seeks the most cost-effective solution. Cluster-based VM migration will be used in the future for more efficient outcomes.Yu et al. (2022)

This study demonstrates the direct importance of the framework structure, tasks, and resources. The primary goal of the proposed approach (LBMPSO) is to properly schedule all incoming tasks to available VMs in order to decrease makespan and enhance machine utilization in cloud computing. Each job must be assigned to a single VM. This approach can reduce total makespan time, boost the use of resources, and balance the load in each virtual machine. To handle the problem of load balancing and job scheduling, a modified PSO algorithm known as LBMPSO is developed in this study. The LBMPSO task scheduling approach is based on the PSO algorithm, which employs a fitness function to determine the optimal particle arrangement. The fitness function computes the execution times of each VM and returns the most elevated execution time as the PSO particle's fitness value (F). The results of the tests show that, under each scenario, LBMPSO reduces the makespan time and increases resource usage when compared to PSOBTS, L-PSO, and DLBA methods. Furthermore, as the number of tasks grows, so does the resource use in all circumstances. The proposed LBMPSO is executed on Eclipse Java Programming Environment and CloudSim toolkit.Pradhan and Bisoy (2022)

The author's goal in this research is to build long-term load balancing of cloud data centers while also delivering efficient external service performance. The data center is often positioned far away from the end users. Distributed servers are cloud environment components that may be accessed using multiple internet hosting programs. Effective cloud node scheduling and load balancing are required to achieve greater Quality of Service (QoS) and efficient operation of external services. This work offers QMPSO, a novel approach for dynamic load balancing across virtual machines based on a hybridization of modified Particle Swarm Optimization (MPSO) with an enhanced Q-learning algorithm. The suggested approach balances the load by reassigning it to the appropriate VMs based on their fitness values. When compared to distinct techniques such as MPSO and Q-learning, the suggested approach enhances the makespan, throughput, and energy usage during load balancing and effectively minimizes task waiting time.Jena et al. (2022)

The research paper explains in detail the concept of scheduling. The author gives an insight into how scheduling takes place in time-shared and space-shared formats. Along with this, scheduling in the cloud on the basis of Task scheduling which can said as VM level scheduling, and VM scheduling, or named Host Scheduling is explained. As Cloudsim Simulator is used, the parent classes are implemented as VM scheduler and Cloudlet Scheduler. The various algorithms are used to analyze the Burst Time parameter on providing certain priority and turnaround time and the average time of different task scheduling algorithms is provided. Amongst the algorithms used, the author has come to the conclusion that FCFS has many shortcomings with respect to waiting time and turnaround time but the Shortest Job First algorithm performs better.Sahoo et al. (2022)

According to the author of this paper, task scheduling can be defined as minimizing loss of time and maximizing performance. This paper is basically a survey of various task scheduling algorithms, and strategies taking into consideration various parameters. Researchers have conducted their work at a particular point in time, limited by constraints in terms of knowledge, space, and time. The author has considered miscellaneous techniques during scheduling and numerous constraints applied, but given the vastness of cloud computing, researchers have been unable to capture all of its aspects simultaneously.Arunarani et al. (2019)

In this research paper, the author has discussed the problem that arises with on-demand use of cloud resources which mainly has pointed down to task scheduling. To work efficiently in distributing complex and different incoming tasks, the author has tried using meta-heuristic algorithms that have the capability of solving scheduling problems and also tried some hybrid algorithms. This paper also covers a brief detail about various simulation tools widely used for task scheduling. The major gap of using meta-heuristic algorithms and their application to task scheduling has been achieved. The open challenges of current issues mentioned as resource scheduling, and quality of service can worked upon as future work. In all this article has a good approach to studying all the necessary algorithms for task scheduling.Houssein et al. (2021)

In this research paper, the concept of Grid computing is discussed with respect to task scheduling The author has mentioned that the use of scheduling algorithms can be more efficient in scheduling applications. However since the genetic inherit algorithms do not have the efficiency which is required for space problems, therefore a combination of local search algorithms is proposed to overcome this gap. A new algorithm is proposed considering parameters such as the number of missed tasks which shows a decrease in makespan. Although the GELS algorithm has the special behavior of greedy algorithms, it doesn't always move to a solution with a better amount of purpose function directly, but it works by examining existing solutions. Although this proposed genetic algorithm is not completely able to cater to the needs for task execution time, but gives better solution for makespan. ZAKARIA et al. (2021)

In this research paper, the author has an opinion that task scheduling should be done but with efficient energy consumption and enable green computing. The traditional task scheduling algorithms which are used frequently are not enough to have a model for energy consumption. To overcome this issue, the parameter of the makespan should be reduced but it also should cater to the mentioned needs. Hence, a new biologically inspired scheduling algorithm is proposed based on the modified Ant Colony Algorithm ensuing load balancing and enhancing energy consumption. ACO was decided as it governs the optimal use of resources available in data centers. This paper contributes to the proposition of a scheme based on an ant system for scheduling and has used a Cloudsim simulator.Ari et al. (2017)

This research paper gives a comparison study of six metaheuristic algorithms serving to maximise convergence speed with simulation done in Cloudsim. The author gives a diagrammatical representation of metaheuristic techniques of scheduling algorithms and traits such as control parameters, optimization parameters, and problems applied are discussed. This paper helps to get a whole view of which algorithms can be used if one wants to work with the mentioned algorithms. This paper involves future work as hybridization to harness the advantages and increase the efficiency of each algorithms with factors such as resource utilisation cost, makespan and load balancing which can be worked upon. Singh et al. (2021)

2.1 Summary

The literature analysis on load balancing, ALO, ACO, and VM utilization has brought to light critical aspects of optimizing cloud computing environments. While existing research demonstrates effective load-balancing strategies and the utility of nature-inspired algorithms such as ALO and ACO, gaps in understanding their complex applications in specific contexts still exist. This research aims to address a better understanding of complex algorithms with an aspect of reducing execution time while managing resources efficiently. We anticipate adding valuable knowledge to the ongoing discussion on optimizing cloud infrastructures for improved performance and resource utilization by building on the insights obtained from this comprehensive literature review.

3 Methodology

Allocating available resources to tasks or processes inside a computer system is known as resource allocation, and its primary goal is to guarantee that resources are used as efficiently as possible to meet the system's performance targets. In this case, task scheduling is essential for maximizing the use of available resources, particularly the CPU. The scheduler determines when each task will execute, which affects how well the CPU is used. Task scheduling strategies have a big impact on resource allocation decisions. Certain jobs, for instance, may be prioritized by a scheduler; resource allocation methods should make sure that these tasks have access to the resources they need to fulfill their priorities. The system must be able to adjust both task scheduling and resource allocation in response to dynamic changes.

The scheduler and resource allocator must modify their choices as tasks come and go from the system and as resource demands change. The task scheduler in multitasking operating systems chooses the next task to run depending on time-sharing, priority, and other rules. The chosen task is subsequently given CPU time, memory, and other resources via the resource allocator. The task scheduler in multitasking operating systems chooses the next task to run depending on time-sharing, priority, and other rules. The chosen task is subsequently given CPU time, memory, and other resources via the resource allocator. Tasks may be divided among several nodes in distributed computing environments. Resource allocation guarantees that the resources allotted on each node are adequate for each task, while task scheduling determines where each task should be executed.In computer systems, resource allocation and task scheduling are linked activities. To achieve the best possible system performance, responsiveness, and resource usage, these two components must effectively coordinate with one another.

Continuous monitoring and adaptive scaling strategies serve as essential components in this search because they enable organizations to respond dynamically to fluctuating demands while maintaining optimal performance. The following section mentions key methodologies and practices for dynamic scaling in cloud environments. This discussion clarifies the strategies that enable organizations to achieve enhanced application availability, and responsiveness through judicious resource allocation, ranging from real-time metric monitoring to the implementation of auto-scaling groups.

- It is important to continuously monitor a variety of metrics, including network traffic, CPU and memory use, and application-specific performance indicators. Monitoring solutions that measure these variables in real-time are frequently provided by cloud providers.
- Establish preset limits for the metrics that are being tracked. The upper and lower bounds at which scaling measures should be initiated are indicated by these thresholds. Define policies that outline the scaling steps to be done under the metrics and thresholds that have been observed. Scale Down (Downward Vertical Scaling): To prevent over-provisioning, the system lowers the resources assigned to instances when the workload drops.
- Scale Out (Outward Horizontal Scaling): This technique divides the workload among several instances by dynamically adding new ones to the system in response

to rising demand. Scale In (Inward Horizontal Scaling): To conserve resources and cut expenses, extra instances are eliminated when demand declines.

- Auto-scaling groups and related features are provided by numerous cloud providers. When circumstances change, auto-scaling groups automatically modify the number of instances. A desired capacity, minimum and maximum limits, and scalability policies are set up for the group.
- Load balancers divide incoming traffic among the instances when scaling horizontally to guarantee uniform utilization. The load balancer pool can have instances added or removed dynamically.
- Organizations can increase application availability, responsiveness, and cost-effectiveness by implementing dynamic scaling, which guarantees that resources are distributed as efficiently as possible based on demand and usage trends.

3.1 Compared Algorithms

3.1.1 Ant Colony Algorithm

For VM to be allocated using Ant Colony Algorithm, firstly state the goals of the optimization process (e.g., minimizing response time, maximizing resource utilization, minimizing energy consumption) and the VM allocation problem. Draw possible virtual machine allocations as paths or solutions. Every ant will build a solution that matches a specific distribution of virtual machines (VMs) among tasks or services.Make a pheromone matrix to show which VM allocation path is most desirable. Ants update their pheromones according to the caliber of the solutions they create. Better solutions ought to produce more robust pheromone trails. Starting with an empty solution, each ant choose which virtual machines (VMs) to allocate by combining heuristic data and pheromone levels.

Heuristic data may consist of variables like virtual machine capacity, closeness, or resource availability. Ants iteratively choose VMs for tasks as they progress through the VM allocation space, building solutions. Pheromones affect the selection probabilities during the probabilistic construction process. Based on the specified goals, rate the quality of each ant's solution (e.g., response time, resource utilization). Pheromones are left behind by ants on the paths they follow, and the quantity left behind correlates with the quality of the solution. Globally update the pheromone matrix regularly.Blum (2005)

Over time, evaporation is used to lower the pheromone levels. More robust solutions have a greater impact on pheromone levels, which in turn affects the trade-off between exploration and exploitation. Continue doing this until a termination criterion is satisfied or for several iterations. The pheromone matrix directs ants toward better solutions as the algorithm iteratively investigates various VM allocations. Decide which of the ants' best solutions to use as the final VM allocation after a predetermined number of iterations.

3.1.2 Ant Lion Optimisation

Ant lions are well-known for digging conical pits in sandy areas to catch ants. As ants fall down the pit's slope, the ant lion waits at the bottom to capture its prey. The Ant Lion

Optimization algorithm was inspired by this predatory behavior. The algorithm keeps a population of potential solutions, each of which is a candidate solution to the optimization problem. In the context of ALO, solutions are referred to as ants, while potential solutions are referred to as prey. The ant lions represent solutions that are searching for an optimal solution in the search space. Ant lions explore their surroundings by taking random walks and building traps to catch prey. Similarly, in ALO, candidate solutions are perturbed at random to explore the solution space. The fitness of solutions determines their likelihood of attracting other solutions, similar to how ant lions trap other solutions.

ALO, which was inspired by the random walks of ant lions, can be used to explore the resource space for VM allocation. Each solution in the population represents a possible VM allocation configuration. The algorithm dynamically adjusts the VM allocation solutions based on their fitness, which could be a metric like resource utilization, load balancing, or cost-effectiveness. The ability of ALO to balance exploration and exploitation can be useful for determining an optimal distribution of VMs across available resources, ensuring efficient utilization, and preventing over-provisioning or under-provisioning. In this case, the fitness function for ALO could be designed to minimize task execution time. It may take into account factors such as the computational capacity of allocated VMs, data transfer times, and overall system responsiveness. Because of ALO's dynamic nature, adaptive resource allocation is possible, ensuring that tasks are assigned to VMs with sufficient resources to minimize execution time.Gulati et al. (2022)

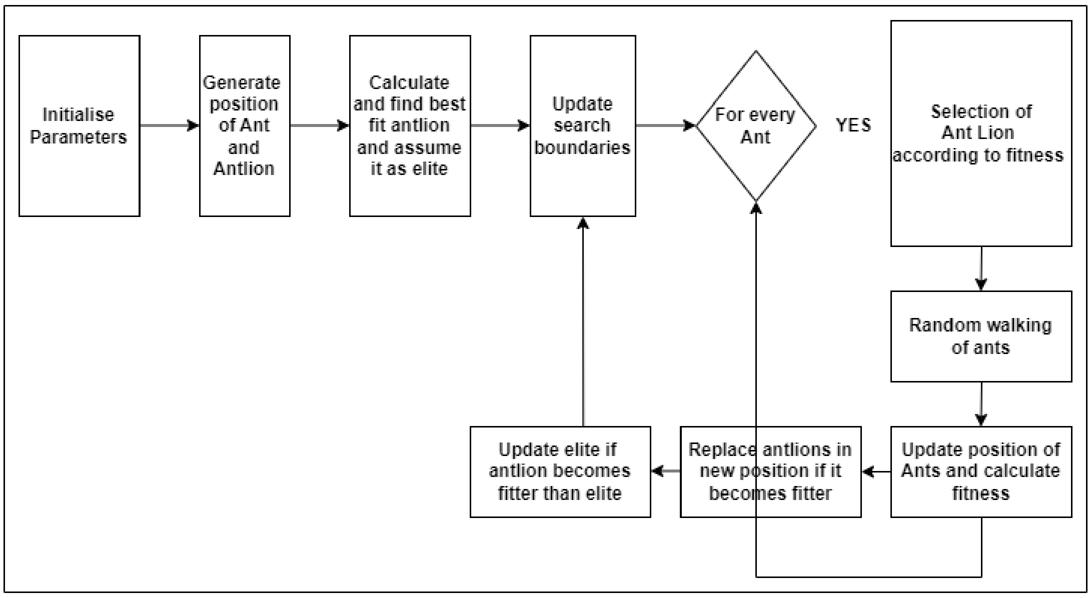
Initialise Parameters
Generate position of Ant and Antlion
Calculate and find best fit antlion and assume it as elite
Update search boundaries
For every Ant
YES
Selection of Ant Lion according to fitness
Random walking of ants
Update position of Ants and calculate fitness
Replace antlions in new position if it becomes fitter
Update elite if antlion becomes fitter than elite

Figure 2: Flowchart of Ant Lion Optimisation

For implementation of algorithms, the Cloudsim simulation tool is used as it gives an insight on how real-world parameters can work. Experiments are carried out by

4 Design Specification

This design specification includes a more precise experimental setup to compare Ant Lion Optimizer (ALO) and Ant Colony Optimization (ACO) in the domain of VM allocation. The research will use a CloudSim simulation environment to get an output in terms of parameters like Task start time, Finish time, VM ID, CloudletID, Datacenter used when the Number of VMs, number of Cloudlets, and number of Datacenters are tuned. A thorough execution time analysis will be performed, measuring the time required for both algorithms to converge. The VM allocation patterns will be visualized and quantified. Multiple runs with varying input scenarios will be executed to ensure robustness. The design will emphasize clarity in algorithm implementation, meticulous data collection, and robust statistics in order to draw meaningful comparisons and derive insights into the relative efficiency of ALO over ACO in dynamic cloud computing.

As mentioned, the implementation is done using Cloudsim. The most important observation for this study is task execution time but to get to that use case other parameters in the Cloudsim file such as the Number of VMs, Number of Cloudlets, Number of data centers, and specifications of each machine should be defined.

Implementation is carried out by experimenting with virtual machines, datacenters, and cloudlets.

- Virtual Machines: A virtual machine is a virtual environment which functions as a virtual computer system with its own CPU, RAM, hostname, and memory on a real hardware system (located off- or on-premises).
- Cloudlets: A cloudlet is similar to the task. Cloudlets frequently support dynamic resource allocation, allowing them to scale resources based on demand. This allows for efficient resource utilization while adapting to varying workloads.
- Datacenters: A datacenter is a centralized facility that occupies computing hardware, networking infrastructure, and storage systems for the management, processing, and storage of data. A datacenter contains VMs according to its holding capacity.

A datacenter is a centralized facility that houses computing hardware, networking infrastructure, and storage systems for the management, processing, and storage of data for various IT services and applications.

5 Implementation

5.1 Simulation Tool

The implementation makes use of JAVA, a versatile and widely used programming language that is compatible with CloudSim, and Python for algorithm execution. The ALO algorithm is expressed in a straightforward and modular manner, ensuring that the fundamental principles of random walks and deterministic movements are effectively captured in the code. Movement probabilities, exploration-exploitation trade-offs, and convergence criteria of the ALO algorithm are carefully parameterized to align with the characteristics of the simulated cloud environment. The implementation is intended to show ALO's scalability and adaptability by demonstrating its ability to dynamically adjust VM allocation based on workload changes and demand fluctuations within the CloudSim environment.

Firstly, number of Cloudlets, number of Vm are defined, after which when that file is executed

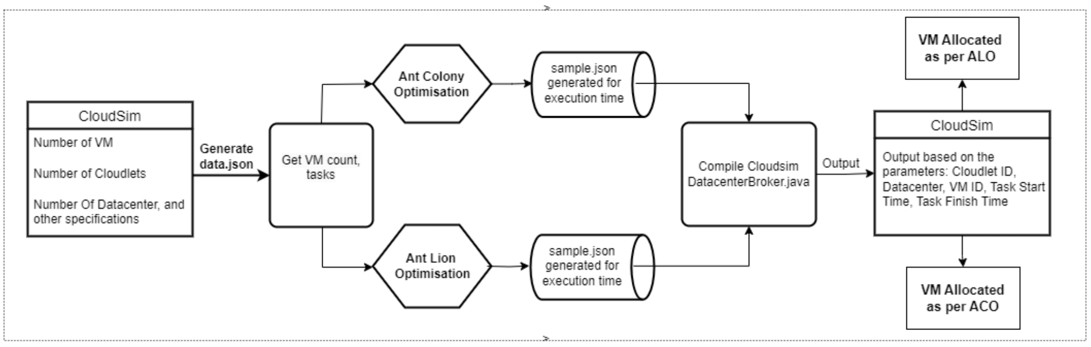
CloudSim
Number of VM
Number of Cloudlets
Number Of Datacenter, and other specifications
Generate data.json
Get VM count, tasks
Ant Colony Optimisation
Ant Lion Optimisation
sample.json generated for execution time
sample.json generated for execution time
Compile Cloudsim DatacenterBroker.java
Output
CloudSim
Output based on the parameters: Cloudlet ID, Datacenter, VM ID, Task Start Time, Task Finish Time
VM Allocated as per ALO
VM Allocated as per ACO

Figure 3: CloudSim Flow for Algorithm Execution

5.2 Comparitive Exploration of ACO and ALO

- Inspiration: ACO is inspired by ant foraging behavior. It is a swarm intelligence algorithm that communicates and finds optimal paths in a solution space using pheromones. ALO was inspired by the hunting behavior of antlions, which build sand traps to catch prey. It focuses on exploring solution space by combining random walks and deterministic movements.

- Exploration and Exploitation: ACO emphasizes the importance of balancing exploration and exploitation. Pheromones guide ants toward promising regions as they explore the solution space probabilistically. Stronger solutions contribute more to pheromone levels over time.ALO uses both random walk and deterministic movements. The random walk allows for exploration, whereas deterministic movements are used to find better solutions. Based on the exploration-exploitation trade-off, the algorithm adapts.

- Solution Representation: ACO has Paths or sequences of decisions are used to represent solutions, and pheromones are associated with these paths. The pheromone levels on paths influence the likelihood that ants will choose those paths. ALO Points in the solution space are frequently used to represent solutions. Antlions explore the environment using a mix of random and deterministic movements, adapting their positions based on the quality of solutions.

- Application to VM Allocation: ACO has been used to solve VM allocation issues in cloud computing. The pheromone matrix guides the exploration of allocation possibilities, with ants representing potential VM allocations. It is frequently used for tasks such as load balancing and resource allocation. While ALO has been used in a variety of optimization problems, its application to VM allocation may be less common. In certain scenarios, the algorithm's adaptability and exploration-exploitation trade-off could be used to allocate VMs.

- Algorithmic Features: ACO uses pheromones for communication and updating solution quality information globally. It's well-suited for problems where decentralized communication among agents is beneficial. ALO combines random walk with deterministic movements. It adapts its behavior based on the random exploration and exploitation of the solution space, potentially offering advantages in terms of convergence.
- Performance and Convergence: ACO is well-known for its ability to find high-quality solutions, particularly to combinatorial optimization problems. Its convergence speed is determined by parameter settings as well as the nature of the problem. ALO's unique combination of random and deterministic movements is intended to balance exploration and exploitation. The algorithm's performance may vary depending on the characteristics of the problem.
- Parameter Sensitivity: The performance of ACO can be affected by parameter settings such as pheromone evaporation rates, exploration rates, and the impact of heuristic information. ALO's performance, like that of many optimization algorithms, can be influenced by parameter settings, and determining appropriate values may necessitate experimentation.
- Maturity and Adoption: ACO is a well-known optimization algorithm with a wide range of applications and a large body of literature. It has been widely used in a variety of fields. ALO is a newer technology, and its adoption and application in various domains, including VM allocation, may be less extensive than ACO.

5.3 Algorithm Implementation

As previously stated, ALO employs the random walking method in a defined search area. Every ant takes a different walk path, which increases the algorithm's exploration. With regard to random walks, the upper bound sand lower bounds of each variable in ant normalises. These walks should concentrate on the ant lion, which describes how ants become trapped in sand pits created by ant lions in nature. The ant lion with the best chance of influencing ant movement is the fittest. To simulate these, a roulette wheel is used. Only two ants, the random ant and the elite ant lion, have the ability to influence the movement of the other ant. The range of the random walks decreases proportionally as the number of iterations increases. Below is the Fitness function Equation derived for this algorithm:Gulati et al. (2022)

- In this phase the random location of the ants is defined.
  This phase is responsible for initialization of basic cloud infrastructure:

  $RWi\& = [RWi^1\& \quad ..RWi^k\& \quad ..RWi^n]$

  The walk of the ant at kth iteration can be defined by below quation.

$NRWi^k = \frac{RWi^k - a}{b-a}$

where RW is the position of individual ant.

- Grab the prey

  This phase defines the boundary limit of each ant LB is the lower bound and UB be the upper bound limit of iteration,

  $LBK = \frac{LB}{D}; \quad UB^* = \frac{UB}{D}$

  $LB_j = AL_j + LB^k$

- Create an ant ambush, it determines new positions for ants which is based on the positions of neighboring ants with the midpoint between lower bounds and upper bounds.
  $Ant_i = ((NRWi, j^k + NRWi, e^k)(UB^k - LB^k) + LBj^k + LBe^k)/2$

- Update fitness function:
  This step is responsible to evaluate the fitness value of the new position of the ant after each iteration where $\alpha + \beta = 1$.

  $Fitnessvalue_i = \alpha * Utilization + \beta * TotalExecutiontime_i$

  $TotalExecutionTime_i = \sum_{i=1}^{n} TaskLength_i / MIPS_j$

6 Evaluation

Efficient Virtual Machine (VM) utilization is critical to the performance, cost-effectiveness, and responsiveness of cloud computing. The Ant Lion Optimizer (ALO) and Ant Colony Optimization (ACO) are compared for VM allocation, with a focus on resource utilization optimization and task execution time. The effective use of computing resources within VMs is measured by VM utilization, which takes into account CPU, memory, and network usage. It is critical to strike a balance between execution time and VM utilization. ALO's dynamic exploration results in adaptive VM allocation, with the goal of maximizing utilization while maximizing execution speed. Graphs depict allocation patterns in a visual manner, revealing algorithmic adaptability to changing workloads.
The inherent adaptability of ALO is expected to result in more efficient VM allocation, resulting in less idle time. CPU usage, memory occupancy, and network throughput are specific metrics that provide insight into dynamic workload adaptability. VM utilization is quantified using quantitative metrics such as the percentage of VM capacity used. The thesis compares ALO and ACO under various conditions, taking task execution time into account. ALO's efficiency in task completion time should complement its advantage in

VM utilization. The comparison of various case studies ensures scalability and consistency in VM utilization patterns.

6.1 Case Study 1

In this experiment, the number of datacenters is 2, the number of virtual machines is 24, and the number of cloudlets which are the tasks are 250. As per the graph, one can visualize that when ACO and ALO are compared the number of VMs used is less in ALO as compared to ACO.

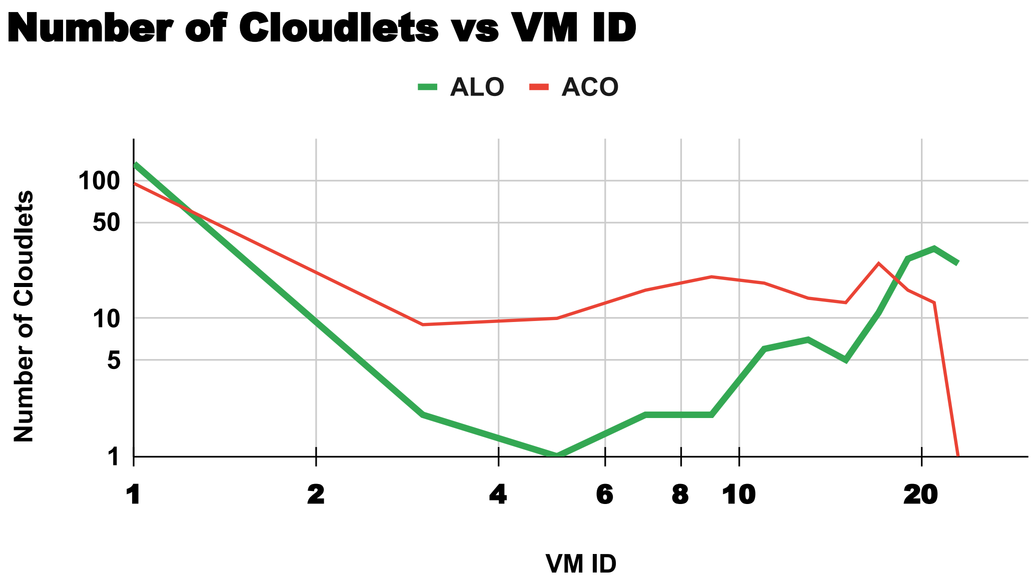

Figure 4: Comparison of ALO and ACO

Due to the limited data volume, the observed trend of fewer virtual machines used in Ant Lion Optimization (ALO) compared to Ant Colony Optimization (ACO) may lack clarity in the given experiment with a smaller scale of 2 datacenters, 24 virtual machines, and 250 cloudlets. With a small task set, the details of ALO's superiority in VM allocation may be difficult to identify. To draw definitive conclusions, the experiment must be repeated with larger datasets, ensuring a more thorough evaluation of ALO's efficiency. The current results point to a potential advantage for ALO, but more testing is needed to draw firm conclusions in a variety of cloud computing scenarios.

6.2 Case Study 2

In this experiment, the number of datacenters is 4, the number of virtual machines is 48, and the number of cloudlets which are the tasks is kept at 500. As per the graph, one can visualise that when ACO and ALO are compared area under the ALO curve is less.

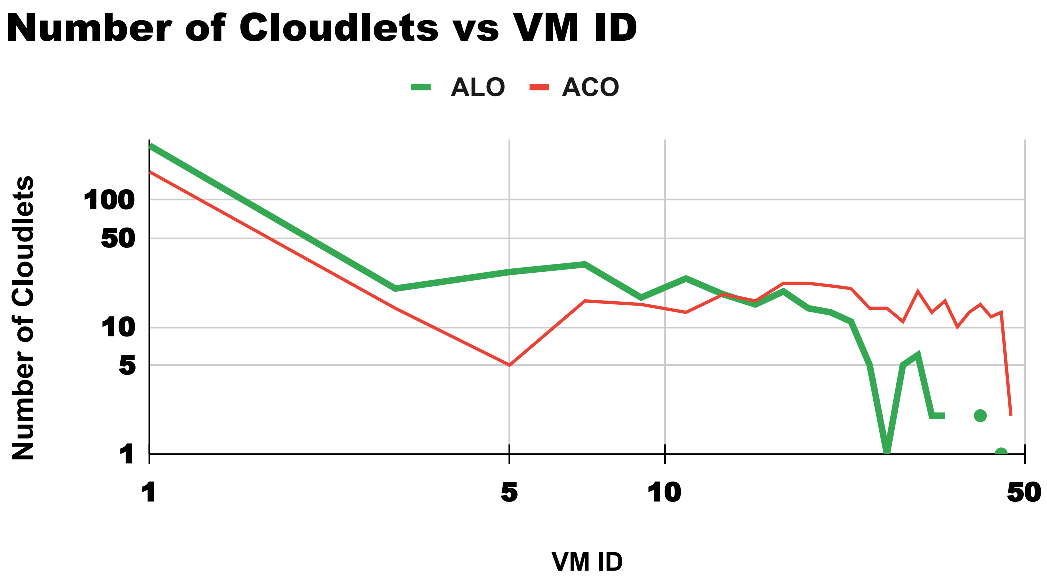

Figure 5: Comparison of ALO and ACO

The experiment's results show that Ant Lion Optimization (ALO) uses fewer virtual machines (VMs) than Ant Colony Optimization (ACO) with 4 datacenters, 48 VMs, and 500 cloudlets when the area under curve concept is used. It suggests that ALO is more efficient in VM allocation. ALO's dynamic approach, which combines random walks and deterministic movements, allows for more precise exploration, which leads to a more optimized allocation strategy. The observed decrease in VM usage demonstrates ALO's ability to avoid overprovisioning, reduce resource redundancy, and emphasize a streamlined allocation process. This result demonstrates ALO's ability to improve resource utilization, lower operational costs, and improve overall performance in cloud environments when compared to ACO.

6.3 Case Study 3

In this experiment, the number of datacenters is 6, the number of virtual machines is 72, and the number of cloudlets which are the tasks are kept at 800. As per the graph, one can visualise that when ACO and ALO are compared the number of VMs used is less in ALO as compared to ACO.

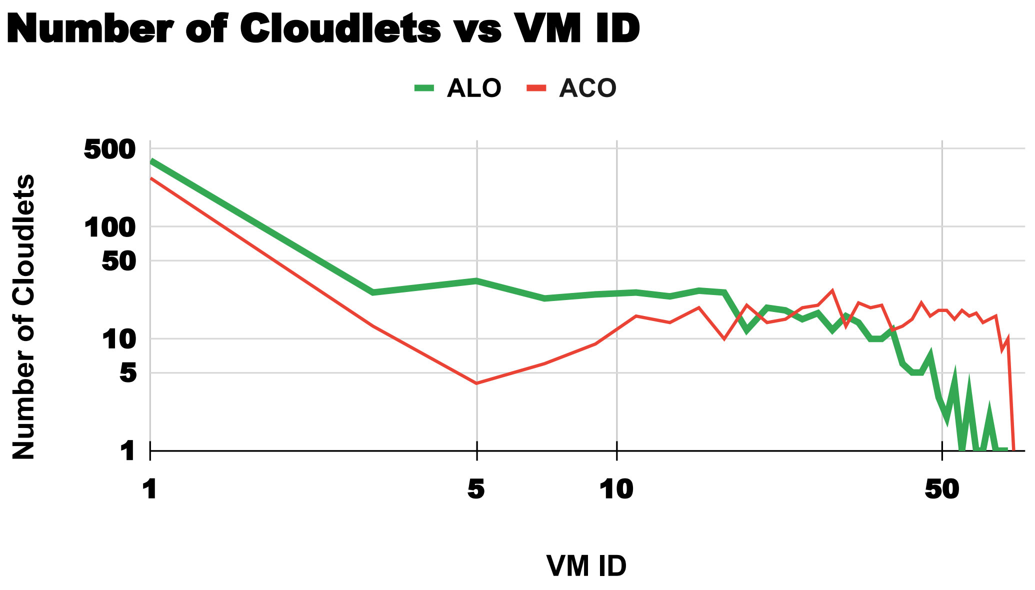

Figure 6: Comparison of ALO and ACO

The experiment's results, which show that Ant Lion Optimization (ALO) uses fewer virtual machines (VMs) than Ant Colony Optimization (ACO), demonstrate ALO's superior efficiency in VM allocation. The dynamic combination of random walks and deterministic movements in ALO allows for adaptive exploration while minimizing resource redundancy. As a result, the VM allocation strategy is streamlined and optimized, demonstrating ALO's ability to achieve more efficient resource utilization. The results suggest that ALO's complex approach to exploration-exploitation trade-offs contributes to its VM allocation superiority, demonstrating potential benefits in terms of lower operational costs and improved overall system performance.

6.4 Case Study 4

In this experiment, the number of datacenters is kept 8, the number of virtual machines is 96, and the number of cloudlets that are the tasks is kept at 1000. As per the graph, one can visualize that when ACO and ALO are compared the number of VMs used are less in ALO as compared to ACO.

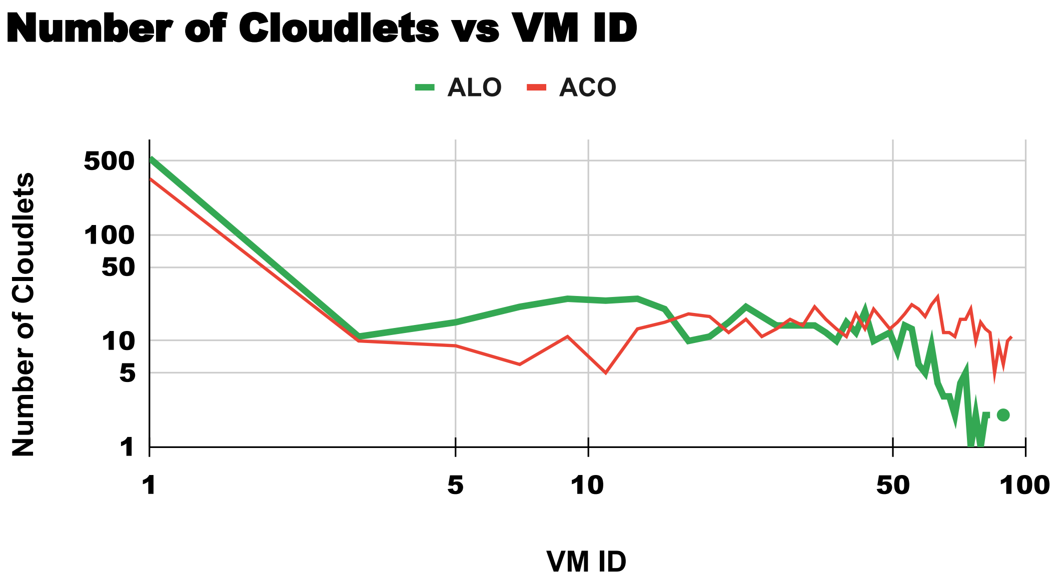

Figure 7: Comparison of ALO and ACO

In the experiment with 8 datacenters, 96 VMs, and 1000 cloudlets, Ant Lion Optimization (ALO) used fewer virtual machines (VMs) than Ant Colony Optimization (ACO). This demonstrates ALO's superior efficiency in VM allocation. The complex combination of random walks and deterministic movements used by ALO allows for a more adaptive and optimized exploration of the solution space, resulting in a streamlined allocation strategy. When compared to ACO, the observed reduction in VM usage highlights ALO's effectiveness in minimizing resource redundancy, avoiding overprovisioning, and ultimately optimizing cloud resource utilization, contributing to potential cost savings and improved system performance.

6.5 Evaluation based on Execution Time

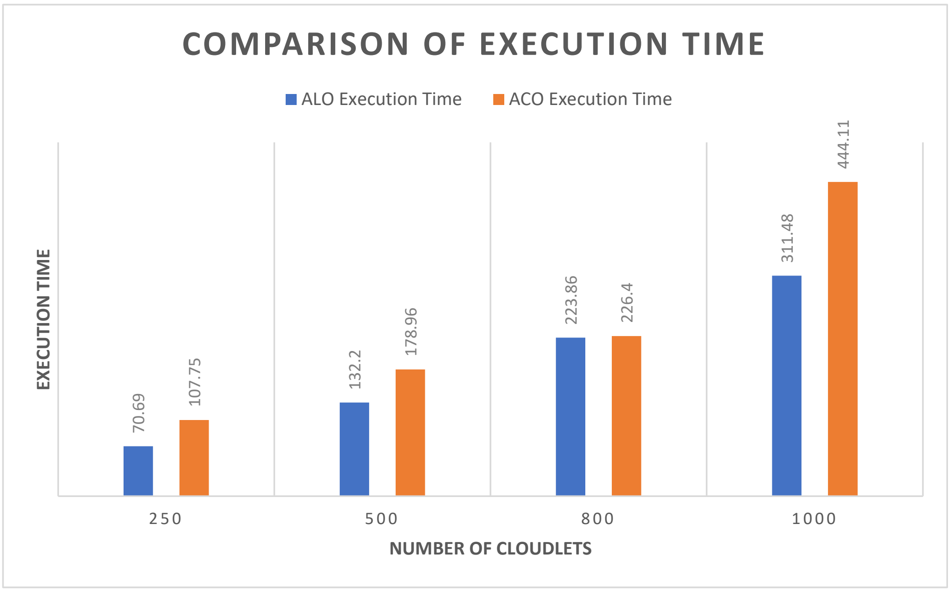

Figure 8: Graph of comparison of Execution Time in ALO and ACO

The bar graph depicting execution times for varying numbers of tasks after applying Ant Lion Optimization (ALO) and Ant Colony Optimization (ACO) shows that ALO consistently outperforms ACO in terms of execution time. This disparity implies that ALO is better at allocating tasks efficiently, resulting in faster task completion. ALO's dynamic exploration and exploitation approach most likely contributes to its superior performance by allowing it to adapt to changing task scenarios effectively. ACO's comparatively longer execution times, on the other hand, may indicate difficulties in optimizing task allocation or adapting to dynamic workloads. These findings highlight ALO's potential for improving task execution efficiency in cloud computing scenarios, indicating it as a viable option for practitioners seeking improved responsiveness and shorter execution times.

6.6 Discussion

The observation that VM utilization is higher with more task execution in Ant Lion Optimizer (ALO) compared to Ant Colony Optimization (ACO) suggests that ALO exhibits a more effective and efficient utilization of virtual machine resources during the execution of tasks. Here are some potential factors and insights that may contribute to this observed difference in VM utilization:

- Adaptability and Dynamic Allocation in ALO: ALO's nature-inspired dynamic exploration behavior may lead to a more adaptive VM allocation strategy. As tasks execute, ALO dynamically adjusts the allocation of VMs based on real-time conditions, optimizing resource utilization to match the evolving workload.

- Parallelism and Faster Convergence in ALO: ALO's parallel nature, where multiple agents explore the solution space concurrently, can lead to faster convergence. The faster convergence may result in more efficient VM allocations, maximizing the utilization of resources throughout the execution of tasks.
- Exploration of Diverse VM Allocation Configurations: ALO's adaptability may allow it to explore a broader range of VM allocation configurations, adapting to different characteristics of tasks. This exploration of diverse configurations can lead to a more fine-tuned and optimized use of VM resources, ultimately contributing to higher utilization.
- ALO's Response to Dynamic Workload Changes: ALO's ability to dynamically respond to changes in workload dynamics could contribute to its superior VM utilization. As the workload fluctuates, ALO may efficiently redistribute VMs to accommodate varying resource demands, ensuring that VMs are utilized optimally.
- Resource-Aware Task Allocation in ALO: ALO's dynamic exploration may involve a more comprehensive consideration of the resource requirements of individual tasks. This resource-aware task allocation could lead to optimized VM assignments, enhancing overall VM utilization throughout the execution process.
- Robustness to Changes in Task Characteristics: ALO's adaptability may make it more robust in responding to changes in task characteristics. As tasks with different resource requirements execute, ALO adjusts VM allocations to ensure optimal utilization, preventing underutilization or overutilization of VM resources.

The inherent parallelism of ALO allows multiple agents to explore simultaneously, contributing to faster convergence. The decentralized nature of ALO allows for independent exploration, potentially utilizing computational resources more efficiently. In scenarios with complex solution spaces, this parallel and decentralized approach can lead to improved execution time efficiency.

In the context of VM allocation, this adaptability may lead to varied and responsive patterns, which are particularly well-suited for dynamic cloud environments due to ALO's nature-inspired dynamic exploration behavior. The dynamic exploration nature of ALO allows it to discover various VM allocation configurations, effectively accommodating fluctuations in workloads. While exploitation is beneficial for stability and convergence to optimal solutions, it may result in less dynamic patterns of VM allocation. Because of ACO's preference for exploitation, established VM allocation paths may be chosen over more adaptive solutions in dynamic environments.

7 Conclusion and Future Work

This study explored the use of the Ant Lion Optimizer (ALO) and Ant Colony Optimization (ACO) algorithms in dynamic resource allocation within the CloudSim simulation framework. The implemented ALO and ACO algorithms have shown efficacy in improving system performance, as evidenced by shorter makespan and better resource utilization.

This work contributes to the existing body of knowledge by demonstrating the adaptability and efficacy of nature-inspired optimization techniques in addressing the challenges posed by dynamic workloads in cloud computing.
However, it is important to recognize the limitations of this study. While CloudSim simulations provide a controlled environment for experimentation, they may not fully capture the complexities of real-world cloud systems. When implemented in live cloud infrastructures with diverse workloads, network latencies, and hardware variations, the performance observed in simulations may differ. The proposed algorithms' scalability must be carefully considered in large-scale cloud deployments.
The difficulties encountered during implementation, such as algorithm parameter tuning and convergence issues, highlight the need for additional refinement and optimization. The obtained results are dependent on specific configurations and assumptions. Future research could address these limitations by conducting experiments in real cloud environments with more diverse and dynamic workloads. Investigating hybrid approaches that combine ALO and ACO with other optimization techniques may improve the robustness and adaptability of resource allocation strategies. In practice, despite its limitations, the findings of this study provide valuable guidance for cloud practitioners and researchers. This work lays the groundwork for the ongoing pursuit of efficient and adaptive resource allocation strategies in the face of dynamic and unpredictable workloads as cloud computing evolves.

References

Afzal, S. and Kavitha, G. (2019). Load balancing in cloud computing–a hierarchical taxonomical classification, *Journal of Cloud Computing* **8**(1): 22.

Ari, A. A. A., Damakoa, I., Titouna, C., Labraoui, N. and Gueroui, A. (2017). Efficient and scalable aco-based task scheduling for green cloud computing environment, *2017 IEEE International Conference on Smart Cloud (SmartCloud)*, IEEE, pp. 66–71.

Arunarani, A., Manjula, D. and Sugumaran, V. (2019). Task scheduling techniques in cloud computing: A literature survey, *Future Generation Computer Systems* **91**: 407–415.

Blum, C. (2005). Ant colony optimization: Introduction and recent trends, *Physics of Life reviews* **2**(4): 353–373.

Gulati, D., Gupta, M., Saini, D. K. and Gupta, P. (2022). Neural inspired ant lion algorithm for resource optimization in cloud, *Sustainable Smart Cities: Theoretical Foundations and Practical Considerations*, Springer, pp. 205–217.

Houssein, E. H., Gad, A. G., Wazery, Y. M. and Suganthan, P. N. (2021). Task scheduling in cloud computing based on meta-heuristics: review, taxonomy, open challenges, and future trends, *Swarm and Evolutionary Computation* **62**: 100841.

Jena, U., Das, P. and Kabat, M. (2022). Hybridization of meta-heuristic algorithm for load balancing in cloud computing environment, *Journal of King Saud University-Computer and Information Sciences* **34**(6): 2332–2342.

Pradhan, A. and Bisoy, S. K. (2022). A novel load balancing technique for cloud computing platform based on pso, *Journal of King Saud University-Computer and Information Sciences* **34**(7): 3988–3995.

Sahoo, B. K., Sardana, A., Solanki, V., Gupta, S. and Saluja, K. (2022). Novel approach of diagnosing significant metrics of load balancing using cloudsim, *2022 10th International Conference on Emerging Trends in Engineering and Technology-Signal and Information Processing (ICETET-SIP-22)*, IEEE, pp. 1–6.

Shafiq, D. A., Jhanjhi, N. Z., Abdullah, A. and Alzain, M. A. (2021). A load balancing algorithm for the data centres to optimize cloud computing applications, *IEEE Access* **9**: 41731–41744.

Singh, H., Tyagi, S., Kumar, P., Gill, S. S. and Buyya, R. (2021). Metaheuristics for scheduling of heterogeneous tasks in cloud computing environments: Analysis, performance evaluation, and future directions, *Simulation Modelling Practice and Theory* **111**: 102353.

Sriram, G. (2022). Challenges of cloud compute load balancing algorithms, *International Research Journal of Modernization in Engineering Technology and Science* **4**(1): 1186–1190.

Yu, D., Ma, Z. and Wang, R. (2022). Efficient smart grid load balancing via fog and cloud computing, *Mathematical Problems in Engineering* **2022**: 1–11.

ZAKARIA, B., ABOUELMEHDI, K., BENI-HSSANE, A. and KHALOUFI, H. (2021). New hybrid algorithm for task scheduling in cloud computing, *Journal of Theoretical and Applied Information Technology* **99**(24).