ISSN: 0974-5823 Vol. 7 No. 2 February, 2022

International Journal of Mechanical Engineering

Energy Efficient and Load Balanced Optimal Resource Allocation framework for Cloud Environment using ML based Meta heuristic techniques

R.KALAIVANI,

Research Scholar, PG Department of Computer science,

CMS College of Science and Commerce, Coimbatore

Dr .V. SUJATHA,

Research supervisor

Vice principal, CMS College of Science and Commerce, Coimbatore

Abstract

Cloud provides resources to the users based on their requirements by using several resource allocation schemes. Reliable resource allocation is one of the major issues of cloud computing. The objective of this paper is to provide an efficient and reliable resource allocation system for cloud environment. The existing research works on resource allocation in cloud mostly address cost and resource utilization whereas this proposed system address the most vital feature, which is cloud data security. The main novelty of this work is to consider not only security but also reduce cost while allocating appropriate resources to the users by using the optimization and machine learning algorithms. The aim of proposed approach is to maximize security while minimizing the cost. In this regard, this research work proposes a Meta heuristic optimization algorithm for resource allocation in cloud. Experimental analysis part provide several performance analyses to validate the proposed approach and the simulation results show that the proposed approach provides increased security while allocating resources to the users.

Keywords: Resource Allocation, Cloud Environment, Machine Learning, Make span

I. Introduction

Cloud computing is a new paradigm in which computing is offered as services rather than physical products. Computing services are provided by third-party service providers which offer consumers affordable and flexible computing services via shared resources. Cloud providers offer different layers/levels of services to consumers over many types of application domains. Server consolidation is a technique for reducing operating costs of computer resources in virtualization. These expenses can lead to incensement of financial costs of servers, power consumption of servers, data-centre cooling systems and labour costs. It is inefficient if there are under-utilized servers that have more space and consume more resources [2].

The cloud computing service models are basically classified into three parts i.e. software as a service (Saas), Platform as a service (Paas) and Infrastructure as a service (Iaas). There are four deployment models followed by Cloud computing i.e. Public Cloud, Private Cloud, Hybrid Cloud and Community Cloud. The proposed system comes under the Iaas which provides memory and processor as a resource. Every user wanted to get the services of cloud as they submit the resource immediately. In this situation, the allocation of resource at real time is a challenging issue. Cloud services are efficiently and optimally allocated to satisfy the requirements of customer. The focus of this paper is detailed and comparative study on different resource allocation strategies by preserving the service level agreement (SLA). The paper brief the algorithms used by the cloud service provider to allocate resources when the multiple resources arrive with different burst time and different resource requirement [2].

The proposed method meets the requirements for effective resource allocation for cloud computing. The effectiveness of the proposed method is evaluated and reveals higher performance in cloud computing.

II. Literature Review

A.V. Karthick et al. proposed a multi queue scheduling algorithm in which resources are divided into clusters of burst time. The resources of equal priority are stored in three queues based on the total burst time of each resource. This algorithm performs better than traditional FCFS, SJF, CBA (combinational backfill algorithm). They can schedule the resources by on demand &reservation category to optimize this algorithm which may return better result based on time. If we add an extra queue which deals with the

resource with highest priority then it would be beneficial for cloud service provider preserving the service level agreement (SLA) [3].

Gulshan Soni et al.,proposed algorithm works with the VM with different configuration. VM is assigned based on the priority of VM to any resource. The calculation of priority is on the basis of memory available processing speed. Resource is assigned toa highest priority VM by load balancer. A unique table is maintains by load balancer with unique Virtual Machine id(Vmid) which is updated on allocation and de-allocation of VM. The resource is not evenly distributed to every VM Solow priority VM may idle for a long time while higher priority VM may be overwhelmed. So the algorithm may be improved which can increase the priority of VM who is idle for a longest time. So that the optimum utilization of the resource can be achieved [4].

Ali Belgacem et al proposed the system for dynamic resource allocation for cloud computing environment. In any case, it faces significant problems as far as service quality, fault tolerance, and energy consumption. It was necessary, at that point, to locate a compelling technique that can viably address these important issues and increase cloud performance. This paper presents a dynamic resource allocation model that can fulfil client need for resources with improved and faster responsiveness. It also proposes a multi-target search algorithm called Spacing Multi-Objective Antlion algorithm (S-MOAL) to limit both the make span and the expense of utilizing virtual machines. In addition, its impact on fault tolerance and energy consumption was contemplated. The simulation revealed that our strategy performed in a way that is better than the PBACO, DCLCA, DSOS and MOGA algorithms, especially as far as make span [5].

Gandhali Upadhye et al., introduced a non-pre-emptive strategy of Nephelescheduler. It computes the critical time for every task. When the execution time reaches to critical point highest expected efficiency task is selected for execution and the current active task is discarded immediately. The expected efficiency of rest of task is recalculated. The task which has smaller expected efficiency than threshold is discarded. So it does not pre-empt the current resource on new resource arrival [6].

Kadda Beghdad et al.,presented Min-Min algorithm for balancing process for all resources with task exchange strategy using scheduling heuristic based on improvement. The heuristic is used to maximize the resources exploitation on cloud computing environments and minimize the total executing time (make span) of task scheduling. The proposed solution considers heterogeneous environment where different number and type of servers used in each cluster is considered for the problem formulation and modelling. A new strategy for task scheduling is proposed in which they collect all resources with help of service request manager and build the cluster of the resources having request of same resource type. So that a specified VMcan be allocated to the particular cluster. There is also a comparison of proposed algorithm with Min-Max algorithm which gives better result [7].

Shridhar G. et al., proposed an optimal load balancing algorithm, which assign a resource to the least loaded VM. It dynamically checks the load of each VM and assigns a resource to the VM which was not assigned in previous iteration. The major finding includes the simulation of algorithm using cloudsim in geographically diversified situation and comparison with load balancing algorithm with active VM. The improvement can be done by taking more dynamic situation of incoming request and response of algorithms in mix (static and dynamic) loads [8].

III. Allocation System Overview

A simple example of multi-objective resource allocation problems is the assignment problem in the distributed client-server Environment. Assume that there are M users that need to be assigned to N servers. Let X is the optimization variable which is a matrix of M ×N whose elements represents clients to be assigned to servers. The objective function is to simultaneously allocate the resource according to the client system request [9]. For optimization

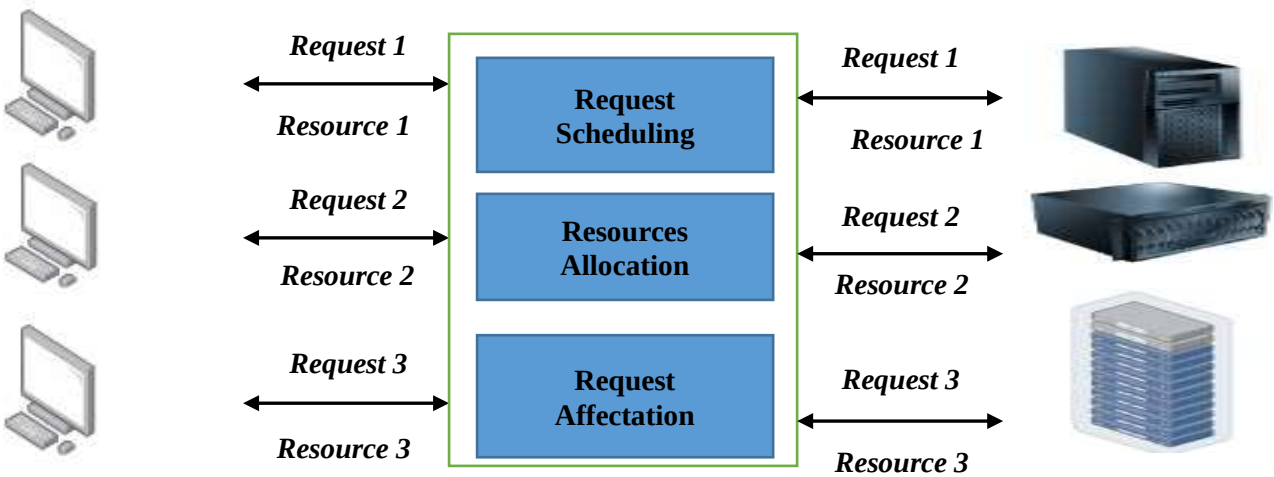


Fig. 1. Allocation System

Figure 1 gives an overview of our scheduling and allocation system. To exemplify our work we can imagine a user connected to a cloud. He pays for its SLAs constraints, according to his budget and his need, and submits his request to be executed. Then, the system deploys the infrastructure, activates computing resources to execute the user request. Each time the system receives a new request, it goes through three phases:

- ***Request scheduling:*** the system selects the request having the highest priorities compromise,
- ***Resources allocation:*** the system decides how many cores must be allocated for a selected request,
- ***Request affectation:*** the system decides which machine executes a selected request.

Finally, when a request is executed and a solution is found, the system collects it from the parallel machines and sends it back to the user [9]. This platform provides a cloud service environment in which developers can use appropriate APIs to make an application such as Facebook, which can be run and shared in anywhere in the world with any platforms without the risk of software pirating. Infrastructure as a Service segment of cloud services provide developing tools with limitless storage and computing powers to developers and ordinary users. For example, Google drive and Apple I Cloud offer cloud storage service for all people including ordinary users and developers.

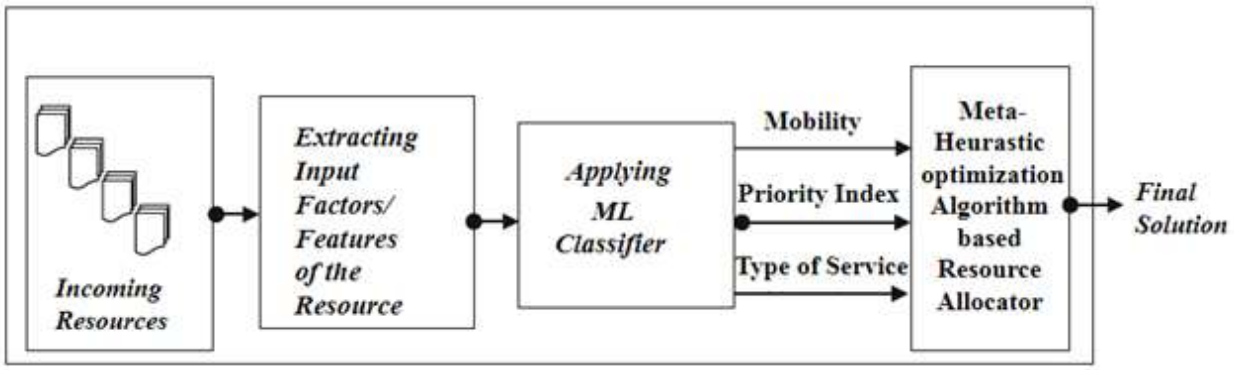


Fig. 2. Proposed system Architecture

Meta Heuristic Scheduling Algorithm - The resources are collected in a set and queued. The process of mapping of events begins in which resources are mapped to a particular virtual machine at predefined time intervals. Meta Heuristic Scheduling Algorithm - In this mode, whenever a resource comes, the resource scheduler assigns it to an available machine. On a cloud, the speed of each processor varies quickly as per the load available at that instance.

IV. Meta-Heuristic Optimization Algorithms

Optimization is to be converted to a more reliable and suitable form of input for the classifier to classify the different features category. This paper utilized the various kinds of optimization algorithm for resource allocation system. Meanwhile the proposed algorithm has been compared to PSO, GA and Whale algorithms in terms of performance of the resource allocation system [10].

1. Genetic Algorithm (GA)

Genetic Algorithm initiates the natural selection and survival of the fittest theories expressed in genetic engineering. Selection process is a process by which the number of features is reduced into a compact subset subsequently bringing high accuracy in classification. The below steps illustrates the genetic algorithm [11].

Step 1: Chromosomes of fixed length are chosen to give a population 'N', the crossover probability 'P_c' and mutation probability 'P_m'.

Step 2: Fitness function of the chromosomes are assessed. Chromosomes with high fitness rate will be chosen, which will mate through reproduction.

Step 3: A primary population $P^{(N)}$ of chromosomes is generated randomly where

$N= x_1, x_2... x_n$ where n = 1, 2….n

If the N value is too high, the algorithm result will not show a conspicuous result. If the N value is very small, the method will not achieve the finest result.

Step 4: Assess the fitness of every respective chromosome in population $P^{(N)}$= $f(x_1)$, $f(x_2)$... f (x_n). A new population $P_n^{(N)}$of chromosomes is generated by a process of natural selection. In deterministic sampling procedure evaluate e(x) = mg(x) for every 'x', in $P^{(N)}$, the virtual fitness is described by g(x).

$$g(x) = \frac{f(x)}{\sum f(x)} x \in P^{(N)}$$

Each fit chromosome x, in the population $P^{(N)}$ which is selected for new population $P_n^{(N)}$, is assigned a fixed number of copies by the integer part of e(x). Chromosomes which are left over are selected for$P_n^{(N)}$, by the fractional values of e(x) from the absolute best values behind there by eliminating chromosomes with minimum fitness and replicate those chromosomes in maximum fitness state.

Step 5: From the available population $P_n^{(N)}$ choose a couple of chromosomes for mating. Parent chromosomes that are fit are chosen with higher probability. Extremely fit chromosomes have a better chance for mating than chromosomes which are fewer fit.

Step 6: Use genetic operators and generate a couple of chromosomal offspring.

Step 7: The chromosomal offspring generated in step 6, are added to the fresh population.

Step 8: Step 5 is reiterated until the new population and primary population equals.

Step 9: The primary chromosomal population is combined by the fresh population.

Step 10: Iterate the method from step 4, till destination target is reached.

GA is a recurring method. Each recurrence is phrased as generation. The generation may range from 50 to 500. After a fixed number of generators the GA is terminated. The simplest chromosome within the population is examined. If the acceptable response is not reached, the process of GA is restarted. In GA bit-string crossover is being used as the main reproducing operator, in which two strings were swapped and two new individual's were formed. GA uses a group of points at a time instead of a single point employed in other optimization strategies

Pseudo code:

Let P(t) and C(t) are parents and offspring in current generation t, respectively and the general implementation structure of GA is described as follows:

Input: Extracted Features, GA parameters

Output: the best solution

Begin

t ← 0;

initialize P(t) by encoding routine;

evaluate P(t) by decoding routine;

while (not terminating condition) do

create C(t) from P(t) by crossover

routine;

create C(t) from P(t) by mutation routine;

evaluate C(t) by decoding routine;

select P(t +1) from P(t) and C(t) by

selection routine;

t ← t +1;

end

output the best solution

end

2. Particle Swarm Optimization (PSO)

PSO is a randomly determined optimization technique copied from flocks of birds or else schools of fish. The flock of birds (swarm) has learnt a co-operative method to discover food and every bird in the swarm, changes the hunt model according to their learning knowledge. The concept of PSO algorithm is related to evolutionary algorithm and swarm artificial life systems [12].

The particles in the swarm (Birds) openly fly over the multidimensional search space. Through the trip, every particle creates its individual velocity along with location. By updating of each particle the entire population is updated. The swarm arrangement drives itself, to move toward the point of upper target function value and in the end the particles assemble around this point. The steps of particle swarm optimization are as follows:

Step 1: Initialization – The swarm particles lie within the pre-defined ranges of velocity and position.

Step 2: Velocity Updating – At every cycle the speeds of the swarm particles are calculated by:

$$\vec{V_i} = W\vec{V_i} + c_1R_1(\vec{P}_{i,best} - \vec{P_i}) + c_2R_2(\vec{g}_{i,best} - \vec{P_i})$$

Where $\vec{P_i}$ = position of particle 'i'

$\vec{V_i}$ = velocity of particle 'i'

$\vec{P}_{i,best}$ = finest position reached by the particle

$\vec{g}_{i,best}$ =best location remembered by the particle individual

'W' = parameter controlling the flying elements

R_1, R_2 =random numbers among 0 and 1

c_1, c2 =cognitive learning factor and social learning factor

The inclusion of variables of each particle gives the PSO, the facility of correctness in searching. The weighing aspects c1, c2 avoid collision among the particles (individuals). After updating particle I, velocity v and random number r is verified also protected in a range indicated, to evade collision.

Step 3:Updating of position – There is an interval among succeeding iterations and hence the positions of the particles undergo change.

$$\vec{P_i} = \vec{P_i} + \vec{V_i}$$

After refreshing, $\vec{P_i}$ must be verified and in the allowable range.

Step 4: Updating of memory – Update $\vec{P}_{i,best}$ and $\vec{g}_{i,best}$ using the formula

$$\vec{P}_{i,best} = \vec{P_i} iff(\vec{P_i}) > f(\vec{P}_{i,best})$$

$$\vec{g}_{i,best} = \vec{g_i} iff(\vec{g_i}) > f(\vec{g}_{i,best})$$

Where ($\vec{x}$) is the point function subject to extension.

Step 5: Destination Checking **–** The technique iterates steps 2 to 4 until definite end states are reached, for a specified number of iterations, when ended. The estimation of $\vec{g}_{i,best}$ and $\vec{P}_{i,best}$ give the result.

The fitness values are not considered in PSO algorithms. This is a big computational advantage over GA, when the population is huge. Arithmetic operation of real numbers is used for calculation of velocity and position. The PSO works on the following principles namely modifying the velocity (accelerating) toward pbest (Personal Best) and lbest (Global Best) locations(local version of PSO) and accelerating towards pbest and lbest locations [22].

Personal Best: The best solution (fitness value) is effectively achieved when individual particle maintains the path of its coordinates in feature space. This value is called pbest.

Global Best:Particle, while maintaining all the features as its topological neighbor, this value is the gbest (global best). When a particle locates with reference to its neighbor it is called lbest.

3. WHALE Optimization Algorithm

Lately there has been developing enthusiasm for WOA which was proposed in [23]. This hunt and advancement calculation is a scientific re-enactment of the conduct and development of humpback whales as they continued looking for food and arrangements. WOA has motivated by the Bubble-net assaulting system, where the whales begin focusing on fish by making winding formed air pockets around their fish down to 12 meters deep from the surface, and afterward, they swim back up to trap and catch their focused on fish. In light of the general places of whales, in this calculation, the investigation procedure is spoken to by the irregular pursuit of food which can be scientifically interpreted by refreshing the old arrangements as opposed to picking the best ones through haphazardly choosing different arrangements. Notwithstanding this intriguing conduct, WOA is quite recognized from other improvement calculations, since it just needs to modify two parameters [13]. These parameters make it conceivable to change easily between both the abuse and investigation forms

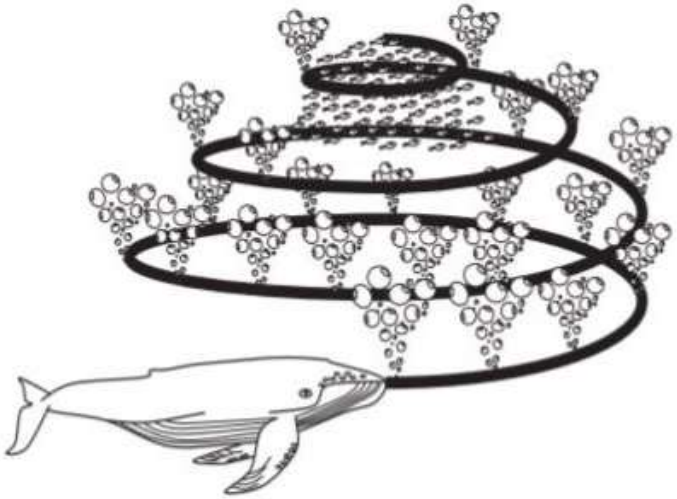


Fig.3. Encircling Attack Prey Searching Methodology for Hump Back Whales

In the following section, we will describe the mathematical model of encircling prey, searching for prey, and spiral bubble-net foraging man oeuvre.

Encircling prey: By the increasing number of iterations from start to a maximum numbers, humpback whales encircle the prey and update their position in the direction of the best search agent. We can mathematically formulate this behavior as:

$$\text{If } (p<0.5 \text{ and } \text{mod}(U) <1)$$

Then the position of the candidate position X (t+1) is updated by the following equations

$$D= \text{mod}\{(C.X)-X(t)\}$$

$$X(t+1) = [X(t) - \{U.D\}]$$

Where p =0.1 (constant) X (t+1) is the best position in the current situation. U and D are calculated by the following equations

$$U= \text{mod}\{2.a.r-a\}$$

$$C=2.r$$

Where a is linearly decreases from 2 to 0 and r is the randomly selected vector

Prey Searching: In prey searching mechanism, X is replaced with the random variables X_{random} and mathematical equation are given as follows

$$D= \text{mod } \{(C.X_{random})-X(t)\}$$

$$X(t+1) = [X_{random}(t) - \{U.D\}]$$

The encircling the prey and spiral updating of the prey has been done during the exploration phase of whale optimization algorithm. The mathematical expression for updation of new position during the spiral process is given by below equation

$$X(t+1) =D^{l}.e^{bl}.\cos(2\pi l) + X^{*}(t)$$

Where D is the distance between the new position and updated position in new generation, b is the constant which varies from the 0 to 1.

V. Proposed system

Suppose that there are m tasks T={t_1,t_2,,...,t_m}, and n virtual machines VM={〖vm〗_1,〖vm〗_2,...,〖vm〗_n}. It is a NP-complete problem which has n^m ways to allocate these tasks to VMs. When a task is bound to a virtual machine 〖vm〗_i the occupation time of 〖vm〗_j depends on the length of task and the CU of the VM [14]. The execution time for t_ij on vmj is given by t_ij

$$t\_ij=L\_i/C\_j$$

where $i \in \{1,2,\ldots,m\}$,

$j \in \{1,2,\ldots,n\}$,

L_(i) represents the length of task t_i,

and C_j denotes the CU (Compute Unit) of virtual machine 〖vm〗_j.

Resource Scheduling Objectives

Designing a framework to implement the scheduling strategy and optimization algorithm for efficient task scheduling.

- Minimizes the make span
- Maximize throughput.
- Minimizes energy consumption.
- Minimize the idle time of the processor.
- Satisfies the quality of service.

Proposed Method

The population is evaluated and selected to create a new generation by the fitness function. To find the correctness of the schedule the fitness function is used:

fitness (P)=min(U_1,U_2,…..,U_n)

where U_1,U_2,…..,U_n are the chromosomes that represent the time taken to finish all the cloudlets execution for their respective assignment .

Algorithm

- Step 1: Start
- Step 2: Calculate the process priority as per their time
- Step 3: Initialize the population as per the process priority
- Step 4: Evaluate the fitness function to determine the fitness of each individual.
- Step 5: Select the fittest chromosome.
- Step 6: Perform crossover mapping over chromosomes by applying SVM algorithm.
- Step 7: Perform mutation by changing the genes of individual parents.
- Step 8: Add the chromosome to a new population
- Step 9: Repeat steps 2 to 7 for all new arrival process
- Step 10: Exit

Cloudlet

Cloudlet in Cloudsim defined the workload, which is to be executed during the simulation run of the cloudsim simulation engine. Cloudlet in Cloudsim is a model class that exists inside the package 'org.cloudbus.cloudsim'. Cloudlet is one of the most important models which defined the specifications for a simulation engine corresponding to the real-life candidate application to be considered for moving to a Cloud-based system. In contrast, we can also define this cloudlet as a single process or task being executed on the cloud-based system which is to be simulated through a Cloudsim simulation engine [15].

Cloud Broker or Datacenter Broker

This class model a broker, which is responsible for mediating negotiations between SaaS and Cloud providers and such negotiations are driven by QoS requirements. The broker class acts on behalf of applications. Its prime role is to query the CIS to discover suitable resources/services and undertakes negotiations for the allocation of resources/services that can fulfill the application's QoS needs. This class must be extended for evaluating and testing custom brokering policies [15].

Steps for Resource Scheduling

1. Create a container to store the VMs which can be passed to a broker later.
2. Pass VM Parameters.
3. Create a container to store the cloudlet.
4. Pass cloudlet parameters.
5. Create Data Centers.
6. Create Broker.
7. Create an Initial Population.
8. Calculate the fitness index of the chromosome that is calculated by the time required forthe completion of each task.
9. Perform Crossover between the different chromosomes gene's lists with SVM algorithm.
10. Perform Mutations in the gene's list.
11. Calculate the fitness function and replace the old population by new population
12. Repeat Step 9-11 until the stopping criteria condition is met.
13. Take the chromosome that has the highest fitness index.
14. Separate the final VM list and cloudlet list from the chromosome.
15. Pass the list to the broker for simulation.
16. Simulation will be finished after every cloudlet are assigned to VMs and finished their execution.

VI. Result and Discussion

Simulation

Simulation is a demand for cloud computing to behold the implementation of real-time outlines. There are various tools available for it such as Cloudsim, CloudAnalyst, iCanCloud, GroudSim, NetworkSim, SPECI, and Cloud Report. We are using Cloudsim and Cloud Analyst in this research project.

CloudSim

Cloudsim is a software which is used to perform modeling, simulation, and experiments in cloud computing services and infrastructures. "The tool is developed by Cloud Computing and Distributed Systems Laboratory at the University of Melbourne." This research using it to implement different resource scheduling algorithms such as First Come First Serve algorithm, Round Robin algorithm, Shortest Resource First algorithm, and Mainly Genetic algorithm. It can instantiate many data centers which

consist of storage servers and physical host machines. These machines host multiple VMs executing several cloudlets. CloudSim can perform simulations of assigning and executing a workload on a cloud infrastructure.

Algorithms	Number of VMs	Start Time	Finish time	Resource utilization time
PSO	10	0.1	1.21	1.11
	20	1.21	2.62	1.42
	30	2.4	4	1.6
	40	1.72	4.34	2.62
Whale	10	0.1	1.24	1.34
	20	1.1	2.47	1.37
	30	2.47	4.12	1.65
	40	2.86	4.7	1.84
GA	10	1.52	2.84	1.32
	20	2.84	4.16	1.32
	30	1.32	5.48	1.32
	40	6.81	8.13	1.32
GA-SVM	10	0.1	1.20	1.10
	20	0.1	1.42	1.32
	30	1.45	2.71	1.26
	40	1.59	2.51	1.28

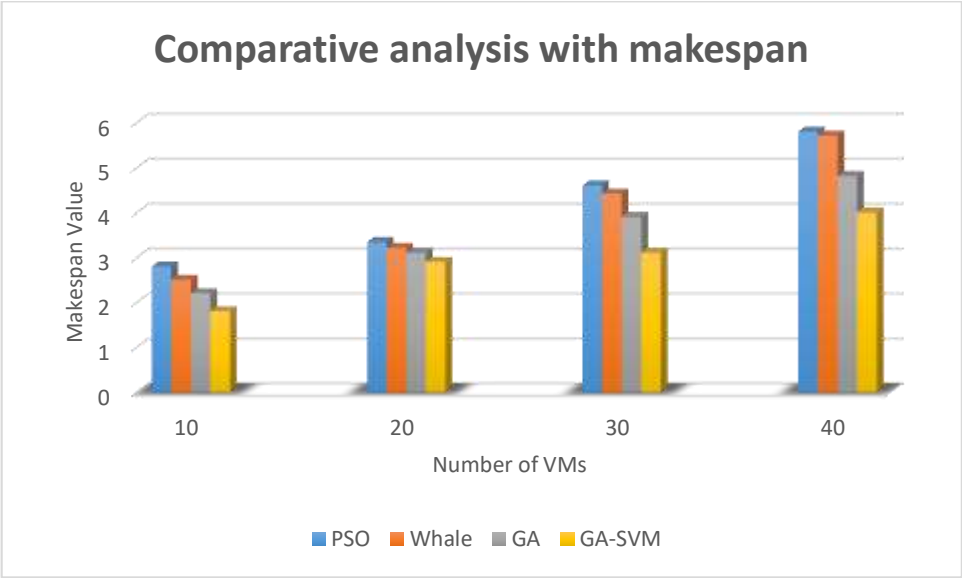
Performance Analysis

i.MakeSpan

Makespan means maximum completion time taken by the cloud system to complete a task. To complete the task T_i, the time E_{ji} needed by the resource R_j and PE_j denotes the total time taken by resource R_j to complete all the tasks. Makespan is computed by below equation [16].

$$F(E) = \max\left\{V_{j=1}^{n} E_j\right\}$$

Where Ej denotes completion of task j.The lesser the makespan, the better the efficiency of the scheduling algorithm. The main aim of the proposed systemis to minimize makespan value. Here, we compare various existing algorithms such as PSO, Whale, GA and our proposed method to evaluate the performance of themakespan value.

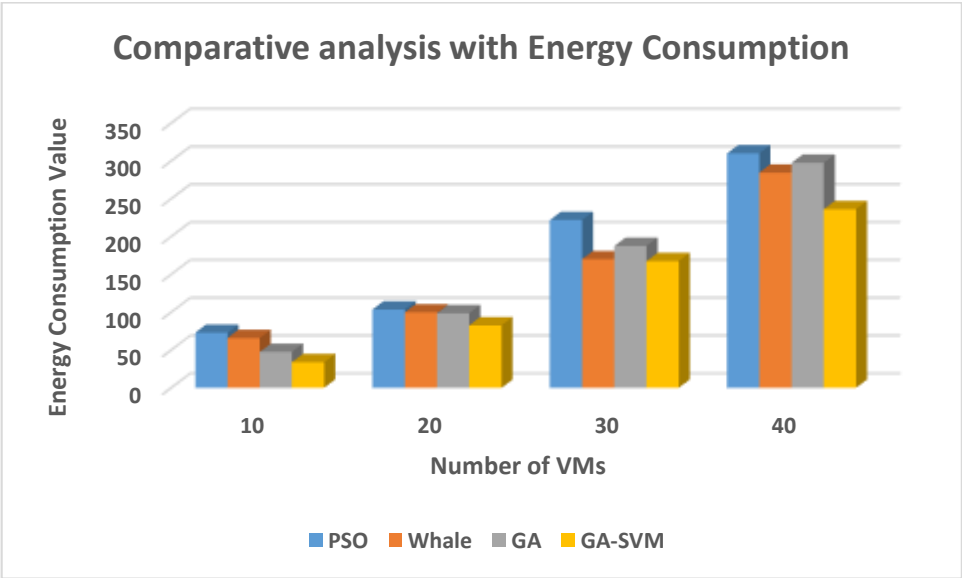


Comparative analysis with makespan

Above figure computes makespan values by comparing the existing algorithms with the proposedalgorithm for 10, 20, 30 and 40VMs. Our proposed method gives 1.8, 2.9, 3.1, and4.0 makespan values which are considered to be minimal compared to the existing methods. Thus, resultsacquired by proposed make it more standard for usage because of the significant difference it makes with makespan whencompared to the remaining algorithms. The above results indicate that proposed method provides better performance.

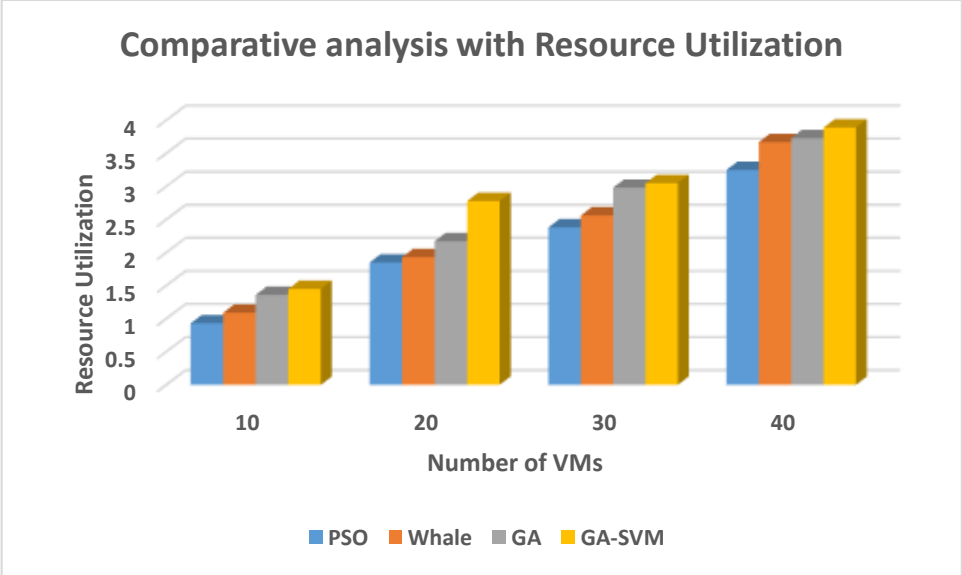
ii.Energy consumption

The below figure shows the comparison of energyconsumption graph, as the number of tasks increases to 10, 20, 30 and 40. The consumption of energy by proposed algorithm is lower when compare to other existing algorithms [16].



iii.Resource utilization

Below figure shows the average resource utilization of the PSO, Whale, GA and Proposed algorithms. The resource utilization is an important metric in the scheduling task process and is beneficial to the cloud provider. The objective is to maximize and achieve a high utilization of resources. Additionally, it is used to keep the resources as busy as possible in order to maximize profit. From the results, it can be shown that the resource utilization of our proposed algorithm is maintained at a high level, which means that it has the best resource utilization compared with the two other algorithms [17].



CONCLUSION

With the booming requests of cloud services today, there are issues of providing on-demand services and resources in a cloud environment and can't be overlooked. Our work offers a persuasive of the time-saving method. The online heuristic resource scheduling algorithm called as a Genetic Algorithm is conducted as experimental studies. We tried to balance the workload by arranging VM based on their processing power and arranging the cloudlets according to their Length The list of VM and cloudlets is then submitted to the broker for the allocation. Broker allocates through a Genetic Algorithm and allocation of resources is done. In this paper, we did a comparative analysis of the results of resource scheduling algorithms like PSO algorithm, Whale optimization Algorithm. The results of using the genetic algorithm are load balancing, improved processor utilization, make span minimization, cost minimization, and maximized throughput. The Genetic algorithm gives better results as compared to the batch heuristic algorithms.

References

1. C. K. Rath, P. Biswal and S. S. Suar, "Dynamic Task Scheduling with Load Balancing using Genetic Algorithm," 2018 International Conference on Information Technology (ICIT), Bhubaneswar, India, 2018, pp. 91-95.
2. M. Lagwal and N. Bhardwaj, "Load balancing in cloud computing using genetic algorithm," 2017 International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, 2017, pp. 560-565.
3. A. V. Karthick, E. Ramaraj and R. G. Subramanian, "An Efficient Multi Queue Job Scheduling for Cloud Computing," 2014 World Congress on Computing and Communication Technologies, 2014, pp. 164-166.
4. G. Soni and M. Kalra, "A novel approach for load balancing in cloud data center," 2014 IEEE International Advance Computing Conference (IACC), 2014, pp. 807-812.
5. Belgacem, A., Beghdad-Bey, K., Nacer, H. et al. Efficient dynamic resource allocation method for cloud computing environment. Cluster Comput 23, 2871–2889 (2020).
6. Anshulika and L. A. Bewoor, "A genetic algorithm approach for solving a resource shop scheduling problem," 2017 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, 2017, pp. 1-6.
7. H. A. Makasarwala and P. Hazari, "Using genetic algorithm for load balancing in cloud computing," 2016 8th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), Ploiesti, 2016, pp. 1-6.
8. S. Yin, P. Ke and L. Tao, "An improved genetic algorithm for task scheduling in cloud computing," 2018 13th IEEE Conference on Industrial Electronics and Applications (ICIEA), Wuhan, 2018, pp. 526-530.
9. F. Fakhfakh, H. H. Kacem and A. H. Kacem, "Simulation tools for cloud computing: A survey and comparative study," 2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS), Wuhan, 2017, pp. 221-226.
10. L. Zhao, Y. Dong and C. Huang, "A Study of Link Load Balancing Based on Improved Genetic Algorithm," 2013 Sixth International Symposium on Computational Intelligence and Design, Hangzhou, 2013, pp. 277-280.
11. Bhosale KK, Jadhav VD. A review of genetic algorithm for metal cutting processes and a research agenda. International Journal for Research in Applied Science & Engineering Technology (IJRASET). 2015;3(V).
12. Durairaj M, Kannan P. Improvised Genetic Approach for an Effective Resource Allocation in Cloud Infrastructure. International Journal of Computer Science and Information Technologies. 2015;6(4):4037-46.
13. S. S. Rajput and V. S. Kushwah, "A Genetic Based Improved Load Balanced Min-Min Task Scheduling Algorithm for Load Balancing in Cloud Computing," 2016 8th International Conference on Computational Intelligence and Communication Networks (CICN), Tehri, 2016, pp. 677-681.