Performance Analysis of Load Balancing Algorithms in Cloud Computing

Rajeev Kumar
Assistant Professor Department of Information Technology, DAVIET,Jalandhar
Punjab, India

Tanya Prashar
Research Scholar of M.Tech (C.S.E), DAVIET
Jalandhar, Punjab, India

ABSTRACT
Cloud computing is a business oriented IT-technology, which is composed of multiple computing technologies accessed via internet. With the rapid increase in cloud usage, it becomes a challenge to deliver the cloud services effectively and efficiently to the cloud consumers on the pay-per usage basis. In this concern Balancing of load has become one of the essential components for the cloud computing environment to perform the effective operations. Scheduling of virtual machines or data centers has to be done properly by using an appropriate load balancing technique. Hence, several algorithms have been developed to process the client's request towards the cloud nodes.. In this present work, a hybridized swarm intelligence technique is proposed to evenly distribute the incoming task requests among the virtual machines or server. Additionally, the performance analysis has been performed using the CloudAnalyst simulator. This paper gives a comprehensive performance analysis of the proposed approach and compares its results with existing Round Robin (RR), Equally Spread Current Execution (ESCE) and ant colony optimization (ACO) techniques. Simulation results have demonstrated that the proposed technique shows a significant outcome in terms of response time, data center processing time and total cost in cloud computing.

General Terms
Algorithms Used: Priority based Bee Colony for Scheduling, For Load Balancing Ant Colony Optimization.

Keywords
Load Balancing, Cloud Computing, Priority based Bee Colony, ACO.

1. INTRODUCTION
Cloud Computing is an IT- Service, where infrastructure, software applications and platform is served over the internet on request, according to the requirement of the users at a particular time. This term is basically used in the case of the internet. The fundamental need of cloud computing is to share and provide the computational resources viz: virtual machines (VMs) as services on demand. Allocating a better VM on user's request is being performed using the load balancing techniques in the cloud computing network. Since the load balancing algorithm plays a vital role while deciding which VM is to be alloted on demand to the client. It is necessary to facilitate and achieve the potentials of computational clouds using an effective scheduling system as to minimize the job execution time. In cloud computing, the problem of scheduling of jobs is an NP-complete. As the number of users associated with the computing increases, the job requests to be scheduled are correspondingly increases, however the existing strategies of task scheduling cannot fulfil its requirements. For such reasons, there is a need of better algorithms for task scheduling for minimizing the computational time and the total processing cost related to that computing. A prominent task scheduling algorithm directly influences the entire cloud system. One of the best examples is a swarm intelligent technique, i.e. bee colony optimization which has been used in this paper for the scheduling purpose. Additionally, load balancing and resource management are the key areas of research in a Cloud Computing environment that is used to evenly dispense the workload on each cloud node, maximizing the resource utilization rate and minimizing the task execution time [1]. In cloud and grid computing, load balancing is a great challenge for dynamic, heterogeneous and complex grid and cloud computing environment. Ant Colony technique is an emerging methodology used for managing, job scheduling and load balancing. Hence, an effective use of this algorithm will lead to maximum usage of resources available, thus improve the throughput and the overall system performance [2] [3]. The main objective of load balancing technique is to uniformly distribute the load among the nodes to optimize the service time of the resources and the response time of the application. Thus ACO technique has been used for load balancing in [4].

The specific contributions of our work involve:

- The novel hybrid approach of load balancing and scheduling of jobs in a cloud computing environment influenced by the foraging behavior of ant colony and honey bee.
- A literature review about several existing load balancing techniques along with their pros and cons.
- Systematic study of the proposed hybrid algorithm using a well defined flowchart depicting its behavioral control structures that how ants and bees foraging behavior inspire scheduling and load balancing for distributed cloud computing system.
- Evaluation and performance analysis, of hybrid technique with respect to other existing load balancing algorithms.

The remaining paper is arranged as: The related works are discussed in Section 2; in Section 3 we discuss the problem formulation. Section 4 provides the brief description of the existing load balancing techniques, Section 5 provides proposed methodology, Section 6 deals with simulation configuration, Section 7 depicts the performance analysis of the algorithms and their comparison. Section 8 will conclude and offers the future work. Section 9 provides the acknowledgement. Section 10 gives the references.

2. RELATED WORKS

In this section, we summarize the load balancing algorithms used in the cloud computing environment in a nutshell. The main focus is on the assignment of all incoming jobs among the available virtual machines with minimal response time. Load balancing is defined as a process of making effective utilization of computational resources by sharing the total workload among the individual nodes of the cloud setup and thereby minimizing the response time of the task.

It has been expressed in [5] that there are many examples of evolutionary algorithms, where genetic algorithm is one of them that is used for scheduling in a network. These algorithms may comprise a memory to retain the last status that helps in reducing the no of agents close to locations in optimal solutions that have been discovered earlier. But unlike swarm intelligence techniques, genetic algorithm don't serve a large number of service consumers. Additionally, the authors in [6] depict that evolutionary algorithms are slower in nature for finding the optimal solutions since there is a need of handling the population movements. The authors in [7] designed algorithms for statically balancing the load on trees, considering that the total load is fixed. The paper [8] proposes an efficient algorithm for data migration in dynamic load balancing by calculating the Lagrange multiplier associated with the Euclidean form of transferred weight. This work can minimize the data movement in homogenous environments in an efficient manner, but it does not support the distributed heterogeneous environments. A novel distributed load balancing technique has been proposed in [9]. The major benefits of the proposed load balancing technique are the distributed structure, less complexity and optimal allocation of virtual machines for each user request. In the studies of [10], VMware based load balancing approach has been offered in order to generate the ants at hop level whenever required, and concurrently the mobile agents memorize every visited node and record their whole information for future reference. A Particle Swarm Optimization (PSO) technique for cloud computing has been developed in [11] which is being inspired by the movement of birds in flocks or fishes in school. In PSO search network, each solution is considered as "particle". Each iteration updates the particle by considering the two "best" values i.e pbest and gbest and these two values are need not be evaluated in ACO algorithm. Unlike PSO technique, ACO reaches with guaranteed convergence to the optimal solution. Moreover ACO is more applicable for problems that require crisp results and PSO is applicable to problems that are fuzzy in nature. The authors in [12] proposed task load balancingbased on foraging nature of bees. In this work a bee nature based algorithm is designed i.e. (HBB-LB) to handle the priority of jobs and reducing the processing time. The proposed technique is more effective and serve less execution time and waiting time as compared to existing load balancing algorithm. However this approach can be further improved by considering the QoS factors.

From the related work it was observed that in most of the studies unlike ACO many algorithms like Genetic algorithm don't serve a large number of clients , and don't create a large number of VMs which the ant colony optimization technique can solve this issue. It has also been noticed that in most of the load balancing algorithms like throttled and round robin most of the time processor remains idle and performance degrades as no. of serve. Further, the two non-preemptive scheduling techniques like First-In-First-Out (FIFO) and RR Policy may lead to load variations between the VMs which results in reducing makespan and flowtime. Thus, such situations may lead to the development of dynamic load balancing algorithms.

3. PROBLEM FORMULATION

The issue of load balancing occurs when the cloud clients try to access and send the request to the same cloud server while other cloud servers don't receive the service request from the cloud clients which leads to the unbalanced workload on the cloud data centers. Therefore, it causes the development of numerous algorithms for scheduling and load balancing. But unlike swarm intelligent algorithm, one of the classes of evolutionary strategies like genetic algorithm does not support multiple users at an instant of time. Many authors have just focused on the availability of nodes and only few factors are taken into consideration like node's memory, processing capacity, etc. Thus we added the some more factors like virtual machine bandwidth, virtual machine computing capacity which is being calculated in respect of millions of instructions per second, no of processors in a virtual machine and image size of a virtual machine therefore all these factors will easily provide the fittest resource for the job to be processed in a cloud. In some research papers of ABC, FCFS priority concept has been considered that could increase the response time. Hence Shortest Job Criteria has been considered to minimize the average response time of the cloudlets. In our approach the antnet concept of forward and backward movement is also taken into account for distributing the workload on the nodes, whereas ABC is applied for searching the optimal path towards the best suitable resource in the cloud network.

4. EXSISTING LOAD BALANCING ALGORITHMS

4.1 Round Robin (RR) Algorithm

The round robin algorithm in the cloud computing is quite similar as the round robin scheduling performed in the process scheduling. This algorithm performs on basis of random choice of the VMs. The datacenter controller (DCC) allots the service calls to a pool of VMs in a cyclic manner. The initial client request is assigned to a randomly selected VM from the group of VMs and then the DCC allots the requests in a round manner. Once the VM is allocated, it is moved to the bottom of the pool of VMs [13]. The major drawback of this type of allocation of tasks is this that it donot favor the advanced load balancing requisites viz: response time for each individual service request and processing time and if the VM is not free then incoming job should wait in the queue.

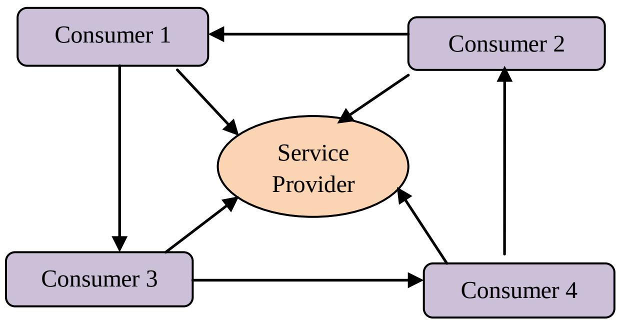


Fig 1: Round Robin Policy

4.2 Equally Spread Current Execution Load (ESCE)

The algorithm of ESCE needs a load balancer that tracks the jobs which are asked for execution. The main requirement of load balancer is to put the tasks in the job pool and assign them to the distinct VMs [14]. The balancer keeps monitoring the job queue frequently for new tasks and then assign them to the pool of free VMs. The load balancer also handles the list of tasks assigned to the virtual servers, which helps them to check which VMs are free and need to be allocated to the new tasks. The experimentation of this algorithm is performed using the cloud analyst simulator. The name itself clearly defines about this algorithm that it works with equally distributing the work load on distinct virtual machines.

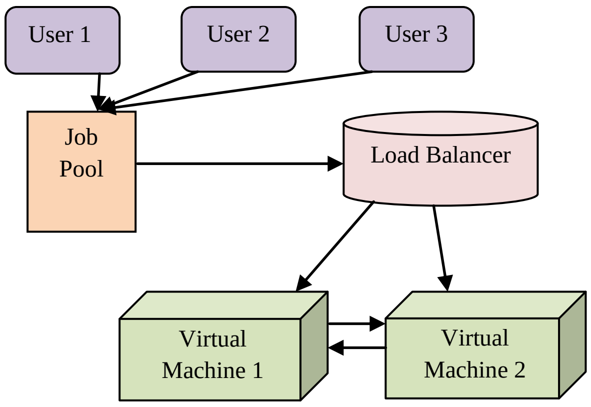


Fig 2: ESCE Algorithm

4.3 ANT COLONY OPTIMIZATION

ACO [15] arises from the way real ants naturally behave. Initially, real ants move randomly in search of food and upon finding the food resource it returns to their colony while laying down pheromones on its path. If new ants also discover such pheromone concentrated track, they will also follow the pheromone trails instead of wandering randomly, return and reinforce it, if they ultimately search the food resource. Thus, when one ant encounters a shorter distance from the ant colony to the food resource, other ants also follow that length due to bio-inspired nature of ants, thus produces a positive feedback i.e. finally makes all the ants to follow a single track. However, if over time ants do not visit a certain path, pheromone trails starts to diminish due to evaporation phenomenon, thus weaken their desirability. The more the time required by an to traverse the path back and forth, the less pheromone trails will be reinforced. From an algorithmic point of view, the pheromone evaporation process is useful to avoid the convergence for a local optimal solution. In our proposed work the ants continuously originate from the randomly selected node so called the head node. These ants travel the entire length and breadth of the cloud network in such a manner that they recognize about the positions of under loaded or overloaded nodes within the cloud system. While traversing, ants will update a pheromone table, which will monitor the resource utilization by each cloud node.

Basic algorithm of ACO is given as:

Step 1: Initialization of ants with pheromones.
Step 2: Locating the ants
Step 3: Selection of next state.
Step 4: Checking of Load Balance
Step 5: Pheromone Updation step.
Step 6: If stopping criteria is met, then stop the execution, else repeat from step 2nd.

4.4 BEE COLONY OPTIMIZATION

The algorithm of artificial bee colony (ABC) is confined to the activities of honey bees for searching the nectar as well as for sharing the information with other bees. In this algorithm, there are generally three types of bees present, i.e. onlooker, bees, scouts and employed bees. The employed bees settle down on the food resource and retain its surroundings in the memory; while onlookers take this information from the employed bees and choose the food resource accordingly. On the other hand the scouts are responsible for discovering the new food resources. The main constituent of the beehive is the dancing area where information is being shared among the bees. The entire communication between bees take place in the dancing region. This information is related to the location and quality of food resources. This dance is known as “the waggle dance”. As information regarding all the optimal food resources is available with the onlookers, that exists on the dance floor, it can choose the most beneficial resource [16]. Given below are the primary steps of the algorithm of bee colony optimization.

Primary algorithm of Bee Colony is given as:

Step 1: Initialization of bee population with their random solutions.
Step 2: Evaluate the fitness function and recruit employed bees.
Step 3: Calculate the fitness function and recruit the onlookers.
Step 4: Move the Scout bees.
Step 5: Evaluate the optimal solution.
Step 6: Check stopping condition, if met, then end the execution, otherwise repeat from 2nd step.

5. PROPOSED SYSTEM

The objective of the proposed work is to make effective scheduling and uniform distribution of workload among cloud virtual resources at an excellent performance rate. This paper works out on two basic swarm intelligent metaheuristic algorithms, including ant colony optimization and priority based ABC. These two algorithms are implemented for scheduling and load balancing within a cloud scenario such that load balancing is performed in order to determine which virtual resource is heavily loaded or underloaded. The goal of this research work is to design and produce an efficient load balancing algorithm within a distributed heterogeneous environment known as cloud. There are no of physical servers known as data centers having multiple virtual machines. These virtual machines are having their own id, no of CPUs, memory, bandwidth and processing power. The proposed algorithm is decentralized to elude bottlenecks and a single failure point. It uses the following parameters:

- No of CPUs,
- MIPS Searching,
- Memory Size of Virtual Machine,
- Bandwidth of Virtual Machine.

The algorithm allows the allocation of priority based tasks on to the virtual machines by considering the constraints of overloading or underloading conditions. Priority has been considered on the basis of Shortest Job First criteria by ~~calculating the no the instructions in a single process. By~~ taking all the above parameters and the overloading and underloading conditions we can get better available virtual machine for the task to be performed on a cloud network. The main objective of the proposed technique is to combine some algorithmic metaheuristic properties of ants and honey bees to develop a novel algorithm for effective scheduling and to uniformly share the workload among the cloud nodes within the cloud scenario by determining the best (shortest) route between nodes in a cloud network. Distance is used to measure the quality of a route therefore shorter the route, more it will be preferred. In this research, ants and bees which are collectively known as mobile agents have the capability to communicate through deposition of stigmergy and through performing the dance, through which they communicates the path to its peers they has just visited. In the implemented ACO algorithm ant is considered as cloudlets composing userbases and food as a virtual machines. In our proposed algorithm after initialization of population ant will continuously originate from the Head node and begins to explore the network.

Hybridized ACO and Priority based Bee Colony algorithm is proposed as:

Begin

Step1: Initialise population with random solutions or it is generated using Cloud Analyst.

Step 2: Initialization of Pheromone Trails.

Step 3: Declare threshold level of nodes (between 0 and 1)

Step 4: While (no of agents! = null) do

Step 5: Prioritize the Population (SJF)

Step 6: Evaluate the fitness of the population of the Bees using given formula:

$$\mathrm{fit}_{ij} = \frac{\sum_{i=1}^{n} \mathrm{cloudlet\_length}_{ij}}{\mathrm{Vm}_j\,\mathrm{\_mips}} \qquad (1)$$

where computing capacity $\mathrm{Vm}_j\mathrm{\_mips}$ is defined by millions of instructions per second for each processor of Vm_j, n is the total no of scout foragers, fit_{ij} defines the fitness function of population of bees (i) for $V\mathrm{m}_j$ or say capacity of $V\mathrm{m}_j$ with ith bee number, cloudlet_length is defined as the task length that has been submitted to $V\mathrm{m}_j$.

The virtual machine (Vm_j) capacity is being calculated using the following parameters

$$Capacity\_Vm_j = Vm_j\,\_cpu * Vm_j\,\_size + Vm_j\,\_bandwidth \qquad (2)$$

where Vm_j_cpu is the total number of processors in a virtual machine Vm_j, Vm_j_size is the virtual machine memory size and Vm_j_bandwidth is the network bandwidth ability of Virtual Machine Vm_j.

Step 7: Try searching for new nodes and select sites for neighborhood search.

Step 8: Calculate the Probability and find the Next Probable Node using the Pheromone values using the given equation (3):

$$\rho_{i,j} = \frac{\left(\tau_{j,k}^{\alpha}\right)\left(\nu_{j,k}^{\beta}\right)}{\sum\left(\tau_{j,k}^{\alpha}\right)\left(\nu_{j,k}^{\beta}\right)} \qquad (3)$$

where $\boldsymbol{\tau}_{j,k}$ is the quantity of pheromone on the path j, k
α is a pheromone control parameter of $\tau_{j,k}$
$\nu_{j,k}$ is the attractiveness of the edge j, k
β is a pheromone control parameter of $\nu_{j,k}$

Step 9: Send bees for the chosen sites and calculate the fitness function.

$$\mathrm{fit}_{ij} = \frac{\sum_{i=1}^{n} \mathrm{cloudlet\_length}_{ij}}{\mathrm{Vm}_j\,\mathrm{\_size}} \qquad (4)$$

Where Vm_j_size is the virtual machine memory size.

Step 10: Select the fittest bee from each patch and select the Node whose Pheromone is highest on the basis of condition i.e. threshold of ACO and fitness value of ABC. At each algorithm iteration, the fittest bee will be chosen to assign tasks in Vm_j

Step 11: Calculate Load Balance Check if the load on selected node less than or greater than threshold Update Foraging or trailing Pheromones using the following equations (5) and (6):

$$\mathrm{FP}(x+1) = (1-\eta_{\mathrm{eva}})\,\mathrm{FP}(x) + \sum_{k=1}^{n} \Delta\mathrm{FP} \qquad (5)$$

where η_{eva} = Evaporation rate of Pheromones,
FP (x) = Foraging pheromones at the edge at time x
FP (x+ 1) = Foraging pheromones at time x+1
ΔFP = newly added Foraging Pheromones.

$$\mathrm{TP}(x+1) = (1-\eta_{\mathrm{eva}})\,\mathrm{TP}(x) + \sum_{k=1}^{n} \Delta\mathrm{TP} \qquad (6)$$

where η_{eva} = Evaporation rate of Pheromones,
TP (x) = Trailing pheromones of the edge at time x
TP (x+ 1) = Trailing pheromones of the edge at time x+1
ΔTP = newly added Trailing Pheromones.

Step 12: Assign remaining bees and remaining ants to search randomly and evaluate their fitnesses and pheromones.

End

The following Fig 3 represents the workflow of the hybrid approach which starts with the initialization of the parameters, following with the construction of ant solution, evaluating the fitness function of bees, calculating the probability value, scheduling of the tasks to the resources, and then balancing of workload is carried out using the pheromone updation technique of the hybrid algorithm.

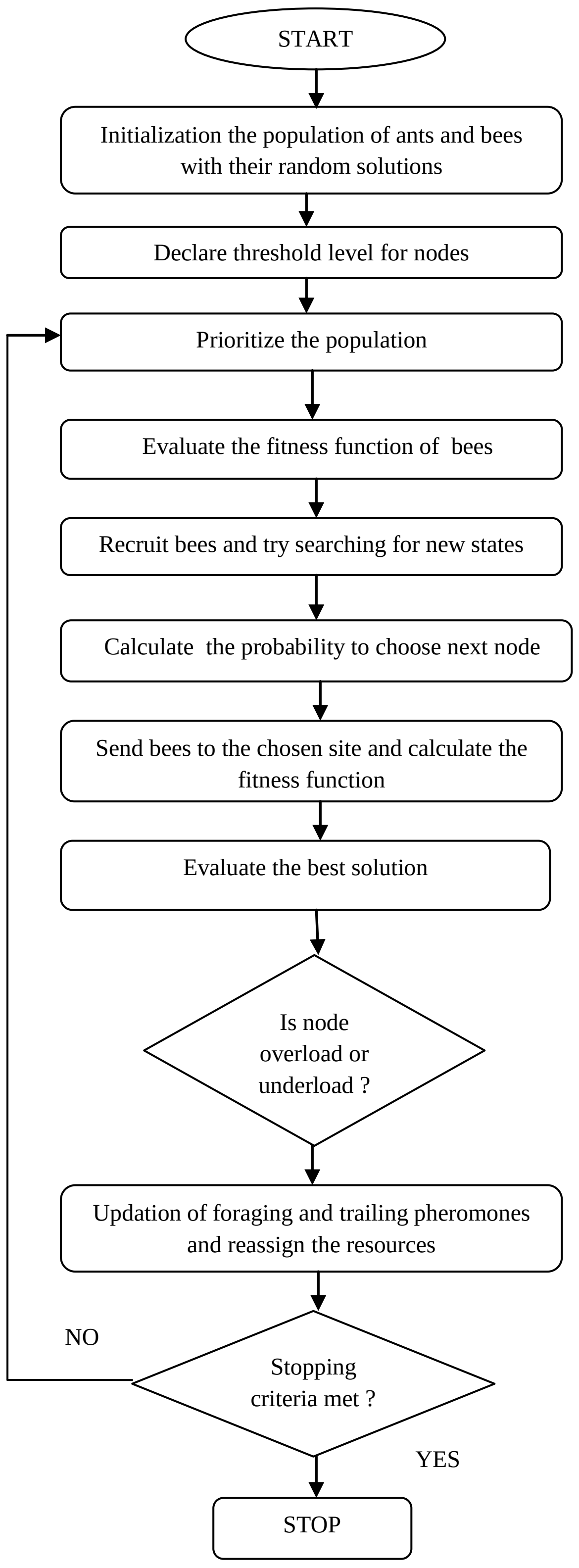

START
Initialization the population of ants and bees with their random solutions
Declare threshold level for nodes
Prioritize the population
Evaluate the fitness function of bees
Recruit bees and try searching for new states
Calculate the probability to choose next node
Send bees to the chosen site and calculate the fitness function
Evaluate the best solution
Is node overload or underload ?
Updation of foraging and trailing pheromones and reassign the resources
NO
Stopping criteria met ?
YES
STOP

Fig 3: Flowchart of Hybrid Algorithm

6. SIMULATION CONFIGURATION

The simulation and performance analysis has been performed using the cloud analyst toolkit by using given specified configuration. It models the cloud data centers, cloud service brokers and resource scheduling policies in [17] but it became necessary to own an easy to access tool with a user friendly GUI that lead to Cloud Analyst tool. The load balancing technique is used by the VmLoadBalancer component of the cloud analyst toolkit. Cloud analyst simulator removes all the programming complexities by designing a user friendly GUI. It allows the user to do parameter sweep experiments. The cloud analyst framework allows setting the regions of cloud based data centers and usersbases. Several parameters can be configured viz: no of user bases, no of requests generated per user per hour, no of VMs, no of cpus, storage amount, bandwidth of the network and other significant parameters as shown in Table 1.

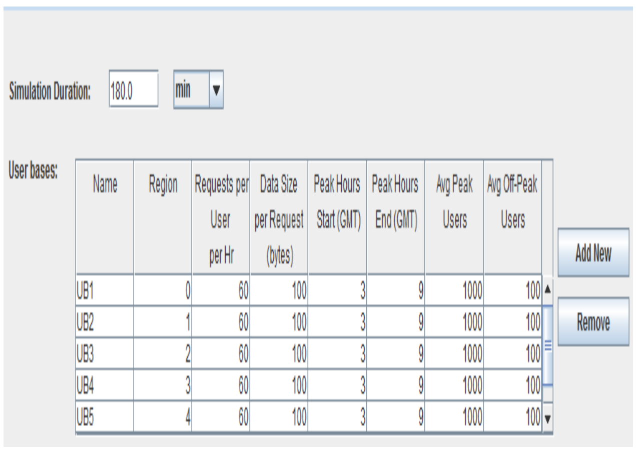
Table 1: Paramter Settings in Simulation

S.no	Parameter	Value
1	VM-Memory	512 MB
2	Data Center OS	Linux
3	Data Center-VMM	Xen
4	Data Center Architecture	x86

On the basis of these parameters, cloud analyst evaluates the simulation and shows its results in a graphical format. Following are the statistical metrics derived as the output of the simulation in the initial version of the simulator:

- Overall Response Time of the system
- Total Data Center Processing Time
- Overall Processing Cost (Sum of Total virtual Machine Cost and Total Data Transfer Cost).

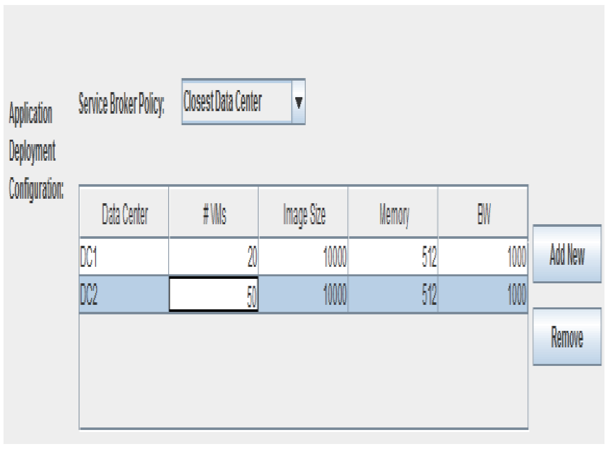
We have defined the parameters for the user base configuration, application deployment configuration and data center configuration as shown in Fig 4, Fig 5 and Fig 6 respectively. As shown in Fig 4 user bases' locations have been set in six distinct regions of the cloud. We have taken two data centers to serve the service request of these user bases. One data center is taken in region 0 and another one in region 2 . On DC1, 20 VMs are allocated and on DC2 there are 50 VMs . The applied load balancing policy is being executed by using the closest data center broker policy where the userbases choose the closest data center to be processed at.


Simulation Duration: 180.0 min

User bases:

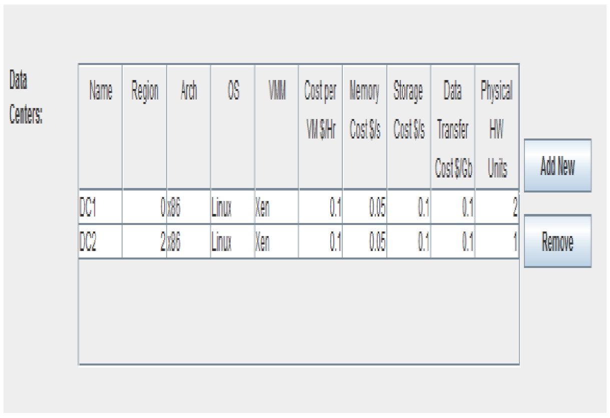
Name	Region	Requests per User per Hr	Data Size per Request (bytes)	Peak Hours Start (GMT)	Peak Hours End (GMT)	Avg Peak Users	Avg Off-Peak Users
UB1	0	60	100	3	9	1000	100
UB2	1	60	100	3	9	1000	100
UB3	2	60	100	3	9	1000	100
UB4	3	60	100	3	9	1000	100
UB5	4	60	100	3	9	1000	100

Fig 4: Configuration of User Bases


Application Deployment Configuration:
Service Broker Policy: Closest Data Center

Data Center	# VMs	Image Size	Memory	BW
DC1	20	10000	512	1000
DC2	50	10000	512	1000

Fig 5: Application Deployment Configuration


Data Centers:

Name	Region	Arch	OS	VMM	Cost per VM $/Hr	Memory Cost $/s	Storage Cost $/s	Data Transfer Cost $/Gb	Physical HW Units
DC1	0	x86	Linux	Xen	0.1	0.05	0.1	0.1	2
DC2	2	x86	Linux	Xen	0.1	0.05	0.1	0.1	1

Fig 6: Configuration of Data Centers

Fig 7 shows the simulation panel during the simulation. It provides the GUI package, i.e. to enter various simulation parameters in a user friendly manner, graphical user interface is provided. The GUI of the cloud analyst toolkit is shown in Fig 7.

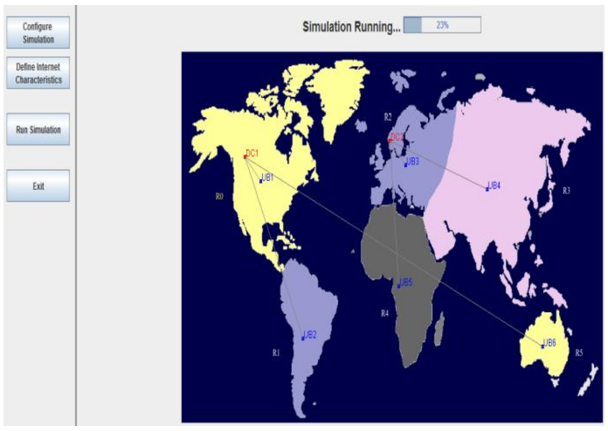


Fig 7 : Simulation Screen of Cloud Analyst

7. RESULT ANALYSIS

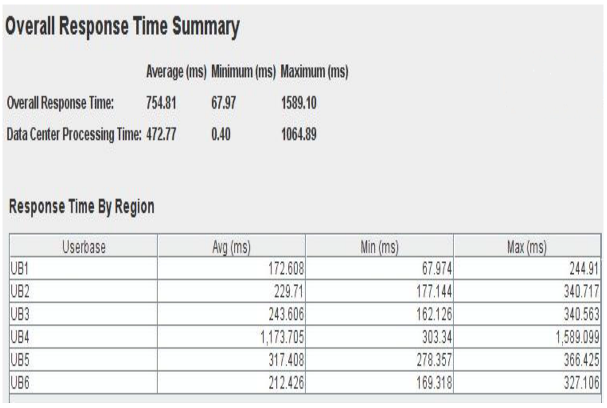
After configuring the simulation parameters and executing the simulation, the results have been computed by the cloud analyst tool as shown in the following figures. We have used the above defined configuration for each loading policy one by one and accordingly the results have been calculated for the metrics like average response time, processing time of data center and total processing cost in serving the client's request.

7.1 Response Time

It is defined as the time interval between the request sent and the response received by the cloud user/ consumer. Overall response time is calculated by the tool for each load balancing policy as shown in the Fig 8, Fig 9, Fig 10 and Fig 11 respectively. As we can see from the following figures average response time of the Round Robin policy and ESCE policy is relatively same while our proposed hybrid algorithm has the significant performance over Round Robin and ESCE techniques and has better values than the implemented ACO algorithm in terms of average response time.

7.2 Data Center Processing Time

For each load balancing algorithm Data Center Processing Time has been calculated by the tool as shown in Fig 8, Fig 9, Fig 10 and Fig 11 respectively. As can be seen from the figures the average data center service time of the proposed technique is improved over ACO algorithm and has a significant improvement over Round Robin and ESCE techniques.

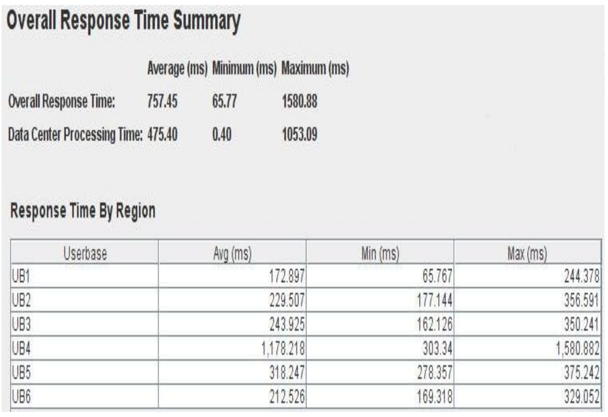

Overall Response Time Summary

	Average (ms)	Minimum (ms)	Maximum (ms)
Overall Response Time:	754.81	67.97	1589.10
Data Center Processing Time:	472.77	0.40	1064.89

Response Time By Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	172.608	67.974	244.91
UB2	229.71	177.144	340.717
UB3	243.606	162.126	340.563
UB4	1,173.705	303.34	1,589.099
UB5	317.408	278.357	366.425
UB6	212.426	169.318	327.106

Fig 8: Response Time and DC Processing Time of Round Robin Algorithm

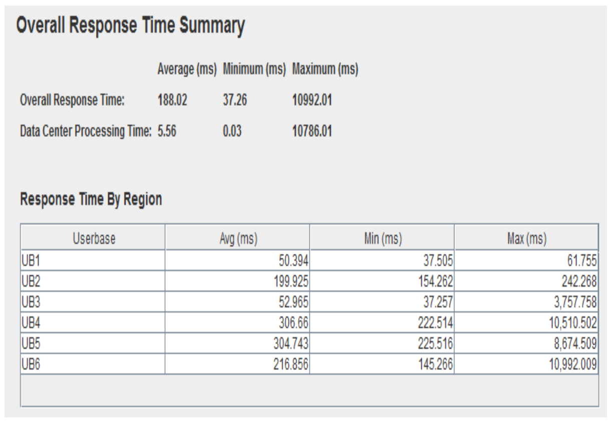

Overall Response Time Summary

	Average (ms)	Minimum (ms)	Maximum (ms)
Overall Response Time:	757.45	65.77	1580.88
Data Center Processing Time:	475.40	0.40	1053.09

Response Time By Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	172.897	65.767	244.378
UB2	229.507	177.144	356.591
UB3	243.925	162.126	350.241
UB4	1,178.218	303.34	1,580.882
UB5	318.247	278.357	375.242
UB6	212.526	169.318	329.052

Fig 9: Response Time and DC Processing Time of ESCE Algorithm.



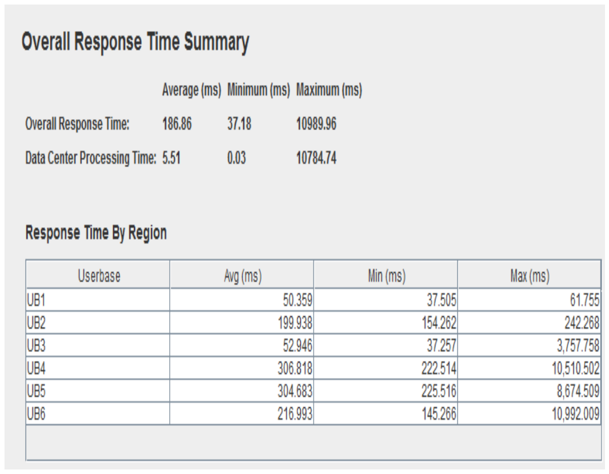
Overall Response Time Summary

	Average (ms)	Minimum (ms)	Maximum (ms)
Overall Response Time:	188.02	37.26	10992.01
Data Center Processing Time:	5.56	0.03	10786.01

Response Time By Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	50.394	37.505	61.755
UB2	199.925	154.262	242.268
UB3	52.965	37.257	3,757.758
UB4	306.66	222.514	10,510.502
UB5	304.743	225.516	8,674.509
UB6	216.856	145.266	10,992.009

Fig 10: Response Time and DC Processing Time of ACO Algorithm



Overall Response Time Summary

	Average (ms)	Minimum (ms)	Maximum (ms)
Overall Response Time:	186.86	37.18	10989.96
Data Center Processing Time:	5.51	0.03	10784.74

Response Time By Region

Userbase	Avg (ms)	Min (ms)	Max (ms)
UB1	50.359	37.505	61.755
UB2	199.938	154.262	242.268
UB3	52.946	37.257	3,757.758
UB4	306.818	222.514	10,510.502
UB5	304.683	225.516	8,674.509
UB6	216.993	145.266	10,992.009

Fig 11: Response Time and DC Processing Time of Proposed Hybrid Algorithm

Table 2: Calculated Response Time of Algorithms

Output Metrics	Load Balancing Algorithms			
	RR	ESCE	ACO	Hybrid
Avg Response Time	754.81	757.45	188.02	186.86
Min Response Time	67.97	65.77	37.26	37.18
Max Response Time	1589.10	1580.88	10992.01	10989.96

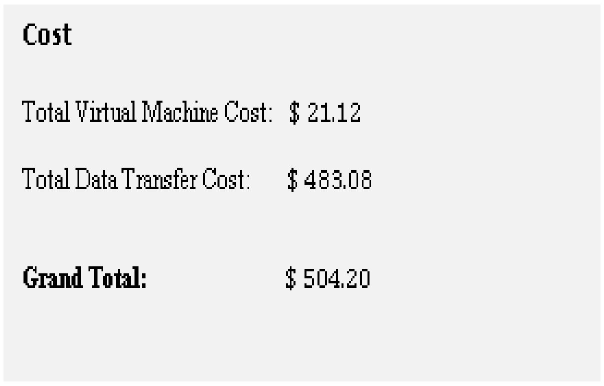
Table 2 and Table 3 shows the evaluated values of the average, minimum and maximum response time of the tasks and processing time of the cloud based data centers for each of the load balancing algorithms, where it is observed that the hybrid approach outperforms the other loading techniques as it performs well in finding the best resource within the cloud network that helps in balancing the load among the resources.

Table 3: Observed DC Processing Time of Algorithms

Output Metrics	Load Balancing Algorithms			
	RR	ESCE	ACO	Hybrid
Avg DC Processing Time	472.77	475.40	5.56	5.51
Min DC Processing Time	0.40	0.40	0.03	0.03
Max DC Processing Time	1064.89	1053.09	10786.01	10784.74

7.3 Total Cost

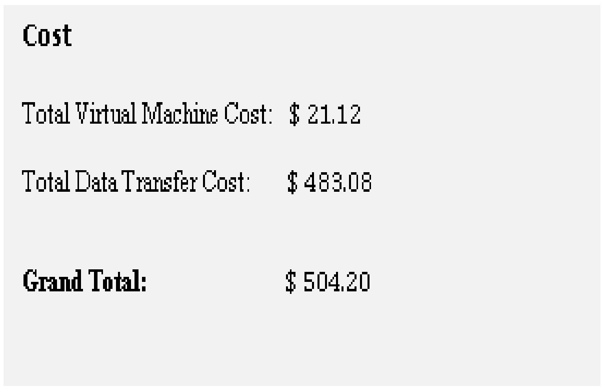
The total processing cost for each load balancing technique is computed by the cloud analyst simulator as shown in Fig 12, Fig 13, Fig 14 and Fig 15 respectively. As seen from the following figures hybrid technique has the far better values than the Round Robin and ESCE algorithms and also outperforms the implemented ACO algorithm and other evolutionary algorithms in terms of cost.



Cost

Total Virtual Machine Cost:	$ 21.12
Total Data Transfer Cost:	$ 483.08
Grand Total:	$ 504.20

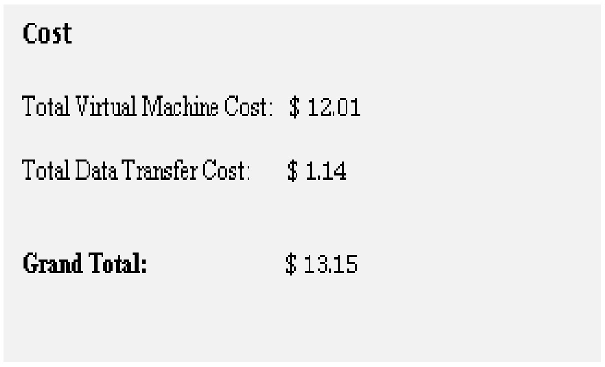
Fig 12: Total Processing Cost of Round Robin Algorithm



Cost

Total Virtual Machine Cost:	$ 21.12
Total Data Transfer Cost:	$ 483.08
Grand Total:	$ 504.20

Fig 13: Total Processing Cost of Equally Spread Current Execution Algorithm

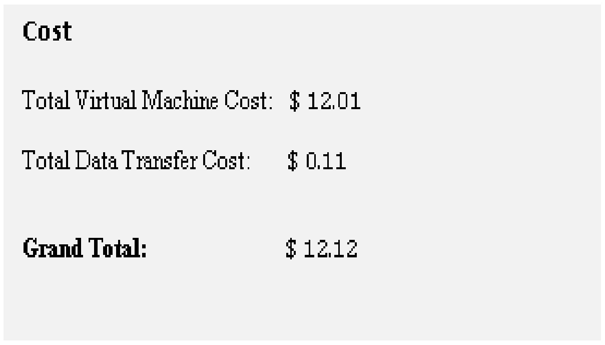

Cost

Total Virtual Machine Cost: $ 12.01

Total Data Transfer Cost: $ 1.14

Grand Total: $ 13.15

Fig 14: Total Processing Cost of ACO Algorithm


Cost

Total Virtual Machine Cost: $ 12.01

Total Data Transfer Cost: $ 0.11

Grand Total: $ 12.12

Fig 15: Total Processing Cost of Proposed Hybrid Algorithm

Table 4: Calculated Cost of Algorithms

Output Metrics	Load Balancing Algorithms			
	RR	**ESCE**	**ACO**	**Hybrid**
Total Cost	504.20	504.20	13.15	12.12

Table 4 shows the observed cost of the algorithms and now we showed the comparison between the resulting response time, data center processing itme and total processing cost for each load balancing algorithm along with the hybrid approach. The difference is shown as in following Table 5 and the graph Fig 16.

Table 5: Result Comparisons of Load Balancing Algorithms

Output Metrics	Load Balancing Algorithms			
	RR	**ESCE**	**ACO**	**Hybrid**
Avg Response Time	754.81	757.45	188.02	186.86
Avg DC Processing Time	472.77	475.40	5.56	5.51
Total Cost	504.20	504.20	13.15	12.12

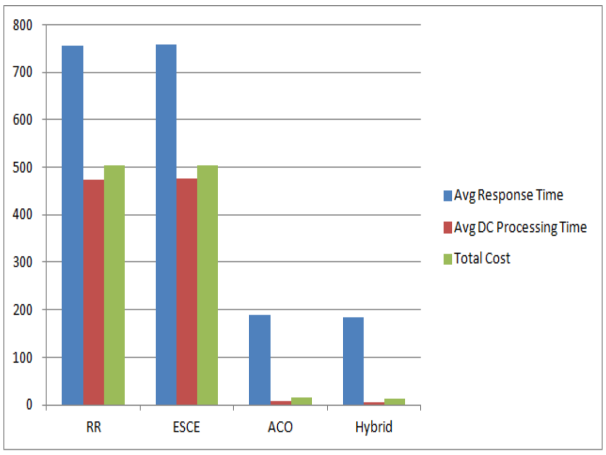


Fig 16: Analytical Compariosn of Load Balancing Algorithms

8. CONCLUSION & FUTURE WORK

Cloud Computing is a widely adopted IT-Oriented service, though there are various existing problems viz: load balancing, migration of Virtual Machines etc. which have not been fully solved till now. Load Balancing is a key issue in the cloud system that is required to dispense the workload in an efficient and scalable manner. It also assures that each computational resource is dispensed evenly and fairly. All the existing algorithms that have been studied mainly considers the reduction of overhead, reducing the migration time and enhancing the performance etc whereas response time is an essential challenge for every engineer to produce the application that can maximize the overall throughput in the cloud scenario. There are many techniques that lack prominent scheduling and load balancing, which leads to increased processing cost. However the proposed algorithm thrives to balance the workload of the cloud infrastructure while reducing the response time for the given number of tasks. The proposed load balancing strategy has been simulated using the CloudAnalyst simulator. Simulation results show that the proposed technique outperformed the existing approaches like RR, ESEC and metaheuristic ACO. Our proposed hybrid technique has a significant improvement over the traditional load balancing algorithms in terms of average response time, average data center processing time, and total cost due to better available virtual machine for scheduling and balancing the workload among them. In contrast, in the hybrid technique the maximum data center processing time has not been improved over other existing algorithms which could otherwise be minimized for best performance. Therefore, this issue can be resolved in our future work. Also in the future we can perform the implementation over a real time cloud setup by considering different load parameters and user requirements.

9. ACKNOWLEDGMENTS

Foremost, I am grateful to the Almighty for the wellbeing that is necessary in completing this research work. I would like to thank my mentor Mr. Rajeev Kumar for his continued support and encouragement extended to me. I wish to express my gratitude to the staff of the Department of Information and Technology for their timely help and support. I would like to express my sincere thanks to Mr A.K Sangal Principal of the institution, for providing me with all the necessary facilities for this research. I am thankful to my parents for the unceasing help encouragement, support and attention. Last but not the least I am dedicating this work to my grandfather Late. Pandit Chunni Lal Shah, whose words of inspiration and encouragement in the pursuit of excellence, still lingers on.

10. REFERENCES

[1] Sowmya Suryadevera, Jaishri Chourasia, Sonam Rathore, Abdul Jhummarwala. (2012). Load Balancing in Computational Grids Using Ant Colony Optimization Algorithm. International Journal of Computer & Communication Technology (IJCCT).

[2] Kumar Nishant, Pratik Sharma, Vishal Krishna, Chhavi Gupta and Kunwar Pratap Singh, Nitin and Ravi Rastogi. 2012 Load Balancing of Nodes in Cloud Using Ant Colony Optimization. 14th International Conference on Modelling and Simulation.

[3] Kuppani Sathish , A Rama Mohan Reddy, (Octobe 2008) Enhanced ant algorithm based load balanced task scheduling in grid computing, IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.10, 21910.

[4] Ratan Mishra and Anant Jaiswal, April 2012 Ant colony Optimization: A Solution of Load balancing in Cloud, International Journal of Web & Semantic Technology (IJWesT) Vol.3, No.2.

[5] A. Y. Zomaya, & Y. H. Teh. (2001). Observations on using genetic algorithms for dynamic load-balancing. IEEE Transaction on Parallel and Distributed Systems, vol. 12, no. 9, pp. 899-911.

[6] Shanti Swaroop Moharana, Rajadeepan D. Ramesh & Digamber Powar. (May 2013) Analysis of Load Balancers in Cloud Computing, In International Journal of Computer Science and Engineering (IJCSE*)*, ISSN 2278-9960 Vol. 2, Issue 2, 101-108 ©IASE.

[7] M. Houle, A. Symnovis and D. Wood, (June 2002). Dimension-exchange algorithms for load balancing on trees, in: Proc. of 9th Int. Colloquium on Structural Information and Communication Complexity, Andros, Greece, pp. 181–196.

[8] Y. Hu, R. Blake and D. Emerson, (1998). An Optimal Migration Algorithm for Dynamic Load Balancing, Concurrency: Practice and Experience 10 , pp 467–483.

[9] Daniel Grosu, Anthony T. Chronopoulos et.al. (2005) 'Noncooperative load balancing in distributed systems', Journal of Parallel and Distributed Computing, Elsevier, Volume 65, Issue 9, pp 1022 – 1034.

[10] M. Salehi & H. Deldari. (2006) Grid Load Balancing using an Echo System of Intelligent Ants, Proceedings of the 24th IASTED International Conference on Parallel and Distributed Computing and Networks, pp. 47-52.

[11] Pandey S, Wu L, Guru S, Buyya R. (2010) A particle swarm optimization-based heuristic for scheduling workflow applications in Cloud Computing environments, In: International conference on advanced information networking and applications (AINA), IEEE Computer Society; p. 400–7.

[12] Dhinesh Babu L.D. and P. Venkata Krishna, 2013 Honey bee behaviour inspired load balancing of tasks in cloud computing environments, Applied Soft Computing, Vol. 13, No. 5, pp. 2292–2303.

[13] Mr.Manan D. Shah, February 2013 Allocation of Virtual Machines In Cloud Computing Using Load Balancing Algorithm, in International Journal of Computer Science and Information Technology & Security (IJCSITS), ISSN: 2249-9555 Vol. 3, No.1.

[14] Ms.Nitika, Ms.Shaveta, Mr. Gaurav Raj, May 2012 Comparative Analysis of Load Balancing Algorithms in Cloud Computing, in International Journal of Advanced Research in Computer Engineering & Technology Volume 1, Issue 3,

[15] Dorigo, M., Gambardella, (1997). L.M.: Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation, 53–66.

[16] Salim Bitam, 2012 Bees Life Algorithm for Job Scheduling in Cloud Computing, Proceedings of The Third International Conference on Communications and Information Technology, pp. 186-191.

[17] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. (June 2009). Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility, Future Generation Computer Systems, Volume 25, And Number 6, ISSN: 0167-739X, Elsevier Science, Amstersdam, The Netherlands, Pages: 599-616.