IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

Examining The Efficiency Of Cloud Computing Load Balancing Algorithms

Kripal Kumar Singh[1]

Research Scholar,MagadhUniversity,Bodhgaya

Dr.Bhola ShankarBarnwal[2]

Assistant Professor,T.S.College,Hisua,Nawada

Abstract

Cloud computing is an IT technology that is focused on business and is made up of several computer technologies that can be accessed online. Delivering cloud services effectively and efficiently to cloud consumers on a pay-per-use basis is becoming more difficult due to the rapid increase in cloud usage. Load balancing has emerged as a crucial element necessary for the cloud computing environment to function efficiently in this regard. Virtual machine or data centre scheduling must be done correctly with the use of an appropriate load balancing technique. In order to handle the client's request towards the cloud nodes, many methods have been created. To distribute the incoming task requests across the virtual machines or server in an equitable manner, a hybridised swarm intelligence technique is proposed in this study. To further enhance the performance study, the CloudAnalyst simulator has been utilised. The suggested strategy is thoroughly examined in this work, and its outcomes are compared to those of the current Round Robin (RR), Equally Spread Current Execution (ESCE), and Ant Colony Optimisation (ACO) algorithms. The suggested technique has a noteworthy impact in terms of reaction time, data centre processing time, and total cost in cloud computing, according to simulation results.

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

Introduction

Cloud computing is an IT service whereby platforms, software, and infrastructure are made available via the internet upon request, based on the needs of the users at that specific moment. Basically, this term refers to the internet. Sharing and supplying computer resources, or virtual machines (VMs), as services on demand is the core requirement of cloud computing. The cloud computing network's load balancing mechanisms are used to assign a better virtual machine (VM) upon user request. Given that the load balancing algorithm is essential in determining which virtual machine (VM) should be assigned to the client based on demand. In order to minimise job execution time, an efficient scheduling system is required to support and realise the potential of computing clouds. The scheduling of jobs is an NP-complete problem in cloud computing. Job requests to be scheduled rise in tandem with the number of users connected to the computer; yet, the task scheduling systems currently in use are unable to meet these demands. The entire cloud system is directly impacted by a well-known job scheduling mechanism. The utilisation of bee colony optimisation, a swarm intelligence technique, for scheduling purposes in this work is among the best instances. Furthermore, the two main study topics in a cloud computing environment are load balancing and resource management, which are utilised to distribute the workload uniformly across each cloud node, maximise resource utilisation, and reduce task execution time [1]. Load balancing in cloud and grid computing presents a significant issue in dynamic, varied, and complex environments. An new approach to job scheduling and load balancing is called the "ant colony technique." As a result, using this technique effectively will maximise the use of the resources at hand, enhancing throughput and system performance as a whole [2] [3]. The primary goal of the load balancing technique is to evenly distribute the load among the nodes in order to maximise the application's response time and the resources' service time. Thus, load balancing using the ACO technique has been implemented in [4].

Our work has made the following specific contributions:

- The unique combination method of job scheduling and load balancing in a cloud computing setting, inspired by the foraging habits of honey bees and ant colonies.
- A overview of the literature on the benefits and drawbacks of the various load balancing methods now in use.
- Thorough examination of the suggested hybrid algorithm with a clear flowchart illustrating its behavioural control structures, which show how the foraging habits of ants and bees inform load balancing and scheduling for distributed cloud computing systems.
- Analysing the hybrid technique's performance in comparison to other load balancing algorithms currently in use.

Relevant Works

We provide a brief overview of the load balancing strategies employed in the cloud computing environment in this section. Assigning all incoming jobs to the available virtual machines with the fastest possible response time is the primary goal. The practice of efficiently using computational resources by distributing the entire workload among the many cloud setup nodes and minimising task response time is known as load balancing.

As noted in [5], there are numerous instances of evolutionary algorithms in use for network scheduling, one of which is the genetic algorithm. In order to reduce the number of agents near areas in optimal solutions that have already been found, these algorithms may include a memory to save the most recent state. Genetic algorithms, however, do not cater to a big number of service users, in contrast to swarm intelligence techniques. Furthermore, because population movements must be managed, the authors of [6] illustrate how evolutionary algorithms naturally take longer to obtain the best answers. Taking into account the fixed total load, the authors in [7] created algorithms for statically balancing the load on trees. By computing the Lagrange multiplier connected to the Euclidean form of transferred weight, the research [8] suggests an effective data migration strategy for dynamic load balancing. This approach does not enable distributed heterogeneous environments, but it can efficiently minimise data transportation in homogeneous environments. [9] has suggested a unique distributed load balancing approach. The suggested load balancing technique has several advantages, including a distributed architecture, reduced complexity, and the best possible virtual machine allocation for every user request. VMware-based load balancing has been proposed in the studies of [10] to produce ants at the hop level as needed, but at the same time the mobile agents memorise each node they visit and log all of their details for further use. In [11], a Particle Swarm Optimisation (PSO) method for cloud computing was created, drawing inspiration from the movement of fish in schools or birds in groups. Every solution in the PSO search network is referred to as a "particle." The two "best" values, pbest and gbest, are taken into account to update the particle throughout each iteration; the ACO method does not necessarily need to analyse these two values. ACO approaches the optimal solution with guaranteed convergence, in contrast to PSO. Additionally, ACO works better for issues requiring precise answers, whereas PSO works better for issues with ambiguous characteristics. Because bees are naturally foragers, the authors of [12] suggested work load balancing. The algorithm used in this work, called HBB-LB, is based on the bee nature and is intended to manage job priorities while cutting down on processing times. Compared to the load balancing algorithm now in use, the suggested method is more efficient and requires less waiting and execution time. Nonetheless, taking the QoS variables into account can enhance this strategy even more.

According to related research, most studies, in contrast to ACO, found that many algorithms, such as the Genetic algorithm, don't service a lot of clients or build a lot of virtual machines (VMs). This problem can be resolved by using the ant colony optimisation technique. Additionally, it has been observed that a lot of the load balancing methods, such as throttled and round robin, cause the CPU to sit idle for the majority of the time, which lowers performance as the number of serves increases. Moreover, load fluctuations across the virtual machines (VMs) may be caused by the two non-preemptive scheduling strategies, such as RR Policy and First-In-First-Out (FIFO), which lowers makespan and flowtime.

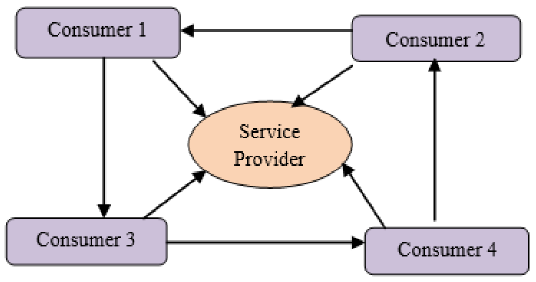
Formulation of Problems

When clients attempt to access and submit requests to the same cloud server, but other servers do not receive the service requests from clients, load balancing becomes an issue and the demand on cloud data centres becomes imbalanced. Consequently, this leads to the creation of multiple load balancing and scheduling techniques. A class of evolutionary techniques, such as the genetic algorithm, does not, however, support several users simultaneously, in contrast to the swarm intelligent algorithm. Several writers have solely addressed node availability, taking into account a small number of parameters such as memory and computing power of the node. In order to provide the best resource for the task to be processed in the cloud, we have added a few more factors, such as virtual machine bandwidth, computing capacity, which is measured in millions of instructions per second, the number of processors in a virtual machine, and virtual machine image size. The FCFS priority idea has been taken into consideration in several ABC research publications, which may result in a longer reaction time. As a result, the Shortest Job Criteria has been taken into account to reduce the cloudlets' average response time.

EXISTING ALGORITHMS FOR LOAD BALANCE

RoundRobin(RR)Algorithm

The round robin algorithm in the cloud computing is quitesimilar as the round robin scheduling performed in the processscheduling.Thisalgorithmperformsonbasisofrandomchoice of the VMs. The datacenter controller (DCC) allots theservice calls to a pool of VMs in a cyclic manner. The initialclient request is assigned to a randomly selected VM from thegroup of VMs and then the DCC allots the requests in a roundmanner. Once the VM isallocated, it is moved to the bottomof the pool of VMs[13].This method of task allocation has a key limitation in that it does not provide the complex load balancing requirements, which include processing times and response times for individual service requests. Additionally, incoming jobs must wait in line if the virtual machine (VM) is not free.



OPTIMIZATION OF ANTCOLONY

Real ants' natural behaviour is the source of ACO [15]. Real ants are first haphazard in their search for food; when they locate it, they return to their colony, leaving pheromone trails in their wake. Should more ants come across a concentrated pheromone trail, they will likewise adhere to it rather than straying aimlessly, and if they eventually seek out food, they will even strengthen it. Because of their bioinspired nature, ants follow shorter routes when they come across them between their colony and the food supply. This leads to a positive feedback loop whereby all the ants eventually follow the same path. Pheromone trails lose some of its appeal due to the evaporation phenomena, though, if ants stop using a particular path over time. Less pheromone trails will be reinforced the longer it takes a to travel the path back and forth. Regarding algorithmic, the pheromone evaporation process helps prevent the convergence towards a local optimal solution. The randomly chosen node, referred to as the head node, is where the ants in our suggested work continuously start. In order to identify the locations of underloaded or overloaded nodes inside the cloud system, these ants travel the full length and width of the cloud network. Ants will update a pheromone table as they travel, tracking how each cloud node is using its resources.

The ACO's basic algorithm is as follows:

Step 1: Ants are first equipped with pheromones.

Step 2: Finding the Ants

Step3: Choose the state to go to next.

Step 4: Verifying the load equilibrium

Step 5: Pheromone Update.

Step 6: Stop the execution if the halting criteria are met; otherwise, go back to step 2 and repeat.

Optimisation of Bee Colonies

The artificial bee colony (ABC) algorithm is limited to the tasks performed by honey bees, such as seeking for nectar and communicating information to other bees. Three different kinds of bees are typically found in this algorithm: onlookers, scouts, and hired bees. The working bees take a seat on the food source and memorise its surroundings; bystanders pick the food resource based on the information gathered by the working bees. The scouts, on the other hand, are in charge of finding new food sources. The dancing area, where the bees exchange information, is the primary component of the beehive. The dance area is where bees communicate with each other in its entirety. The location and calibre of food resources are related to this information. The term "waggle dance" refers to this dance. Because everyone present on the dance floor has access to knowledge about all the best food sources, they can select the most advantageous one [16]. The main steps of the algorithm for optimising bee colonies are listed below.

The Bee Colony primary algorithm is as follows:
Step 1: Using their random solutions, the bee population is initialised.

Step 2: Find employed bees and assess the fitness function.

Step 3: Determine the fitness function and enlist bystander support.

Step 4: Move the Scout bees.

Step 5: Assess the best course of action.

Step 6: Verify the halting condition. If it is satisfied, stop the execution; if not, go back to step 2.

PROPOSED SYSTEM

The suggested work aims to create efficient scheduling and equitable task distribution among cloud virtual resources at a high rate of performance. The two fundamental swarm intelligent metaheuristic algorithms that are worked out in this paper are priority based ABC and ant colony optimisation. These two techniques are used in a cloud environment for scheduling and load balancing, where load balancing is done to identify which virtual resource is underloaded or heavily laden. The purpose of this research project is to create and develop an effective load balancing algorithm for use in cloud computing, a distributed heterogeneous environment. Data centres, which are physical servers housing several virtual computers, do not exist. Each of these virtual computers has an ID, memory, bandwidth, CPU count, and processing power of its own. To avoid bottlenecks and a single point of failure, the suggested algorithm is decentralised. It makes use of the subsequent parameters:

- No of CPUs,
- Searching of MIPS,
- Virtual Machine memory size,
- Virtual Machine Bandwidth.

Finding the best (shortest) path between nodes in a cloud network is the primary goal of the suggested technique, which combines some algorithmic metaheuristic properties of ants and honey bees to create a novel algorithm for efficient scheduling and equitable workload distribution among cloud nodes in the cloud scenario. A route's desired length increases with its distance, as this serves as a gauge of its quality. The ants and bees utilised in this work, known as mobile agents, can deposit stigmergy and dance to communicate with one another and inform their peers of their recent whereabouts. The created ACO algorithm sees ants as cloudlets, which are made up of virtual machines that represent user bases and food. As per our proposed approach, once the population is initialised, ants would continuously begin their network exploration from the Head node.

Proposed is a hybridised ACO and priority-based Bee Colony algorithm.

Start

Step 1: Generate the population using Cloud Analyst or initialise it using random solutions.

Step 2: Pheromone Trails are initialised.

Step 3: Establish the nodes' threshold level (between 0 and 1).

Step 4: While (null) agents are present,

Step 5: Set the population's priorities (SJF)

Step 6: Use the provided formula to assess the population's fitness of bees:

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

Wherein cloudlet_length is defined as the task length that has been submitted to mj, n is the total number of scout foragers, ij defines the fitness function of the population of bees () for mj or, to put it another way, capacity of mj with ith bee number, and computing capacity is defined as millions of instructions per second for each processor of Vmj. Utilising the following parameters, the virtual machine (Vmj) capacity is determined.

$$Capacity\_Vm_j = Vm_j\_cpu * Vm_j\_size + Vm_j\_bandwidth$$

The variables Vmj_size, Vmj_bandwidth, and Vmj_cpu represent the virtual machine's memory size, processor count, and network bandwidth capacity, respectively.

Step 7: Look for new nodes and choose websites for a neighbourhood search.

Step 8: Utilising the pheromone values and the provided equation (3), calculate the probability and determine the next likely node.

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

Where kα is a pheromone control parameter of τj, k vj, k is the attractiveness of the edge j, and kβ is a pheromone control parameter of νj, k is the quantity of pheromone on the path j.

Step 9: Calculate the fitness function and dispatch bees to the selected locations.

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

Where the virtual machine memory size is denoted by Vmj_size.

Step 10: Choose the node with the highest pheromone among the fittest bees in each patch based on the ACO threshold and the ABC fitness value. The fittest bee will be selected to assign jobs in Vmj at each algorithm iteration.

Step 11: Determine the Load Balance Verify if the chosen node's load is greater than or less than the threshold. Revisit Using equations (5) and (6) below, determine if foraging or trailing pheromones:

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

Where ηeva is the pheromone evaporation rate, FP (x) is the foraging pheromones at the edge at time x, FP (x+1) is the foraging pheromones at time x+1, and ΔFP is the freshly added foraging pheromones.

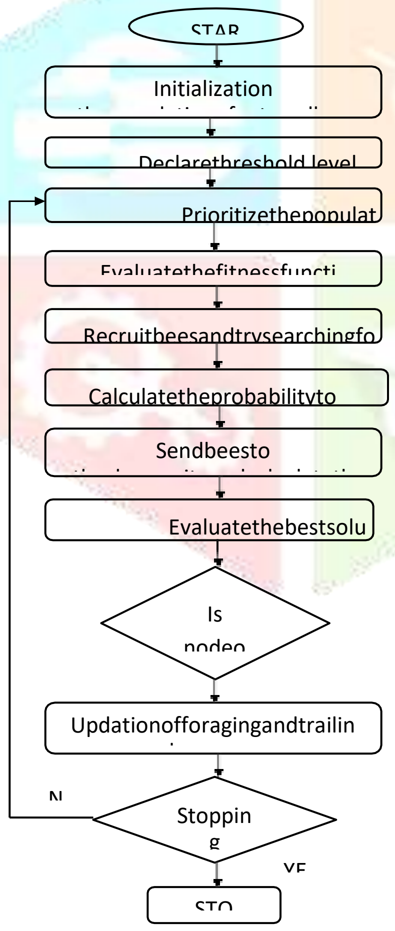
$$\mathrm{TP}(x+1) = (1-\eta_{\mathrm{eva}})\,\mathrm{TP}(x) + \sum_{k=1}^{n} \Delta\mathrm{TP}$$

Where ηeva means the rate at which pheromones evaporate, TP (x) means the trailing pheromones of the edge at time x, TP (x+1) means the trailing pheromones of the edge at time x+1, and ΔTP means the freshly added trailing pheromones.

Step 12: Assign the surviving ants and bees to a random search, then assess their pheromones and fitness levels.

END

The workflow for the hybrid approach is shown in Fig. 3. It begins with parameter initialization, moves on to building the ant solution, assesses the bees' fitness function, computes the probability value, schedules the tasks according to available resources, and uses the hybrid algorithm's pheromone update technique to balance the workload.



, sustainable agriculture may be able to support a portion of this population.

Using the cloud analyst toolbox and the provided settings, the simulation and performance analysis were carried out. It models cloud data centres, cloud service brokers, and resource scheduling policies [17], but the need for a user-friendly, easily accessible tool eventually gave rise to the Cloud Analyst tool. The VmLoadBalancer part of the cloud analyst toolbox employs the load balancing approach. By creating an intuitive graphical user interface, the cloud analyst simulator eliminates any programming complexity. It enables parameter sweep experimentation by the user. Setting the locations of cloud-based data centres and user bases is possible with the help of the cloud analyst framework. As seen in Table 1, a number of factors can be set, including the number of users, the number of requests made by each user in an hour, the number of virtual machines (VMs), the number of CPUs, the quantity of storage, the network bandwidth, and other important characteristics.

S.no	**Parameter**	**Value**
1	VM-Memory	512MB
2	DataCenterOS	Linux
3	DataCenter-VMM	Xen
4	Data CenterArchitecture	x86

These parameters are used by cloud analyst to analyse the simulation and provide the results in a graphical way. The statistical measures that were determined as the simulation's output in the first iteration of the simulator are as follows:

- The system's overall response time,
- Total data centre processing time, and
- Overall processing cost (which is the sum of the costs associated with virtual machines and data transfers) are all included.

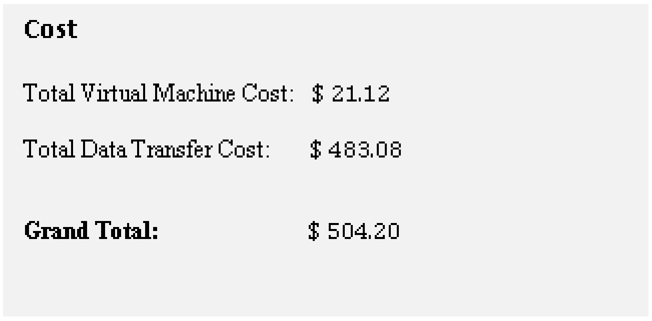
Figures 4, 5, and 6 illustrate the parameters that we have established for the user base configuration, application deployment configuration, and data centre configuration, respectively. User bases are located in six different cloud zones, as Fig. 4 illustrates. To fulfil the service requests of these user populations, we have taken over two data centres. There is a data centre in area 0 and another in region 2. There are allotted 20 virtual machines (VMs) on DC1 and 50 VMs on DC2. The closest data centre broker policy is being used to carry out the applicable load balancing policy, allowing user bases to select the closest data centre for processing.

RESULT

Following the setup of the simulation's settings and its execution, the cloud analyst tool computed the results, which are displayed in the accompanying figures. The above-specified configuration was applied to each loading policy individually, and the results were computed for metrics such as average response time, data centre processing time, and total processing cost in fulfilling the client's request.

Total Cost

The cloud analyst simulator calculates the overall processing cost for each load balancing method, as indicated in Figures 12, 13, 14, and 15, in that order. The hybrid technique performs better than the ESCE and Round Robin algorithms, as well as the implemented ACO algorithm and other evolutionary algorithms in terms of cost, as can be seen in the accompanying figures.

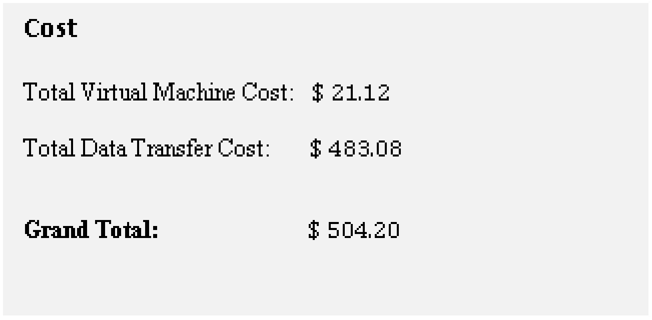

Cost

Total Virtual Machine Cost: $ 21.12

Total Data Transfer Cost: $ 483.08

Grand Total: $ 504.20

Fig 9: Round Robin Algorithm's Total Processing Cost

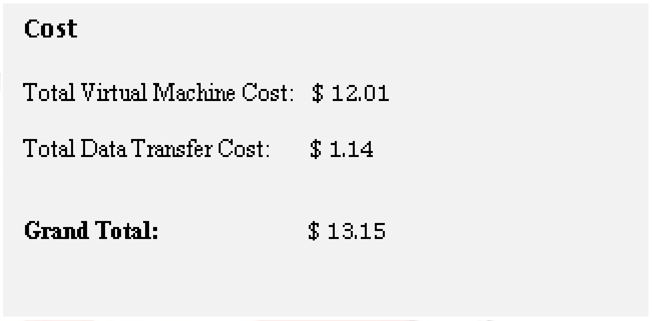

Cost

Total Virtual Machine Cost: $ 21.12

Total Data Transfer Cost: $ 483.08

Grand Total: $ 504.20

Fig 10:An equal spread current algorithm's total processing cost

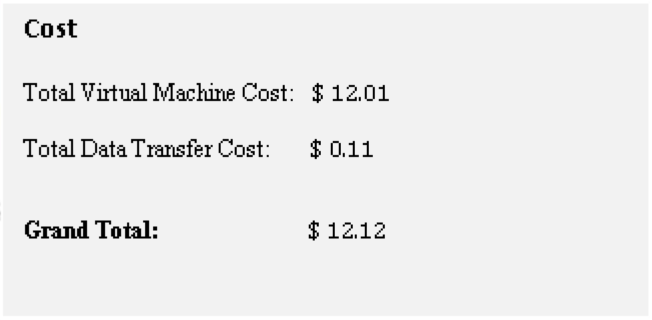

Cost

Total Virtual Machine Cost: $ 12.01

Total Data Transfer Cost: $ 1.14

Grand Total: $ 13.15

Fig 11: Total ACO Algorithm Processing Cost


Cost

Total Virtual Machine Cost: $ 12.01

Total Data Transfer Cost: $ 0.11

Grand Total: $ 12.12

Fig. 15: The proposed hybrid algorithm's total processing cost

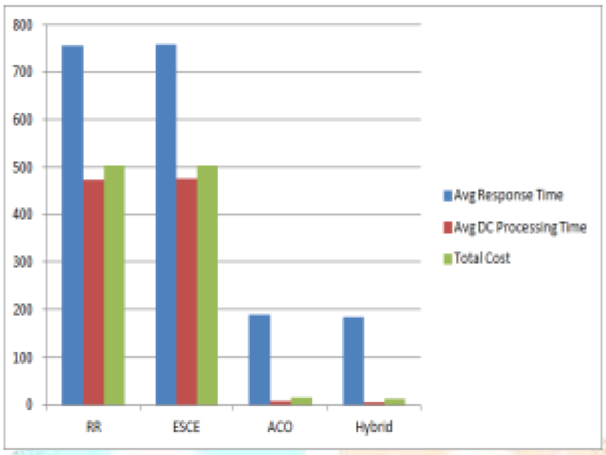
Table4:CalculatedCostofAlgorithms

Output Metrics	LoadBalancingAlgorithms			
	RR	ESCE	ACO	Hybrid
Total Cost	504.20	504.20	13.15	12.12

Table 4 displays the observed cost of the algorithms. Next, we compare the hybrid approach and each load balancing technique in terms of response time, data centre processing time, and total processing cost. The accompanying

Table 5 and the graph Fig. 16 illustrate the difference.

Table5:Comparing Load Balancing Algorithms for Different Results



CONCLUSION & FUTURE WORK

Although cloud computing is a popular IT-oriented service, there are still a number of issues that need to be resolved, such as load balancing and virtual machine migration. A crucial component of the cloud system that is needed to distribute the burden in an effective and scalable way is load balancing. Additionally, it guarantees the equitable and uniform distribution of every computational resource. While response time is a critical challenge for every engineer to create an application that can maximise the overall throughput in the cloud scenario, all of the existing algorithms that have been studied primarily consider the reduction of overhead, reducing the migration time and improving the performance, etc. Numerous approaches lack load balancing and visible scheduling, which raises processing costs. Nonetheless, the suggested technique succeeds in balancing the cloud

OutputMetrics	**LoadBalancingAlgorithms**			
	RR	**ESCE**	**ACO**	**Hybrid**
Avg Response Time	754.81	757.45	188.02	186.86
Avg DCProcessingTime	472.77	475.40	5.56	5.51
TotalCost	504.20	504.20	13.15	12.12

infrastructure's strain while speeding up response times for the specified quantity of tasks. The CloudAnalyst simulator has been used to replicate the suggested load balancing technique. According to simulation results, the suggested method performed better than the state-of-the-art methods like RR, ESEC, and metaheuristic ACO. Because we have superior virtual machines accessible for scheduling and workload balancing among them, our suggested hybrid technique performs significantly better than the old load balancing algorithms in terms of average response time, average data centre processing time, and overall cost. On the other hand, the hybrid technique has not resulted in an improvement over other current algorithms that might minimise the maximum data centre processing time for optimal performance. As a result, we can fix this problem in our next work. We may also execute the solution over a real-time cloud arrangement in the future by taking various load parameters and user requirements into account.

References:

[1] Sowmya Suryadevera, Jaishri Chourasia, Sonam Rathore,AbdulJhummarwala.(2012).LoadBalancinginComputationalGridsUsingAntColonyOptimizationAlgorithm.InternationalJournalofComputer&Communication Technology(IJCCT).

[2] Kumar Nishant, Pratik Sharma, Vishal Krishna, ChhaviGupta and Kunwar Pratap Singh, Nitin and Ravi Rastogi.2012LoadBalancingofNodesinCloudUsingAntColony Optimization. 14th International Conference onModellingandSimulation.

[3] Kuppani Sathish , A Rama Mohan Reddy, (Octobe 2008)Enhancedantalgorithmbasedloadbalancedtaskschedulingingridcomputing,IJCSNSInternationalJournalofComputerScienceandNetworkSecurity,VOL.8 No.10, 21910.

[4] Ratan Mishra and Anant Jaiswal, April 2012 Ant colonyOptimization: A Solution of Load balancing in Cloud,InternationalJournal of Web&SemanticTechnology(IJWesT)Vol.3, No.2.

[5] A.Y.Zomaya,&Y.H.Teh.(2001).Observationsonusinggeneticalgorithmsfordynamicload-balancing.IEEETransaction on Parallel and Distributed Systems, vol. 12,no.9,pp. 899-911.

[6] ShantiSwaroopMoharana,RajadeepanD.Ramesh&DigamberPowar.(May2013)AnalysisofLoadBalancersinCloudComputing,InInternationalJournalofComputer Science and Engineering (IJCSE), ISSN 2278-9960Vol.2, Issue2,101-108©IASE.

[7] M.Houle,A.SymnovisandD.Wood,(June2002).Dimension-exchangealgorithmsforloadbalancing ontrees,in:Proc.of9thInt.ColloquiumonStructuralInformationandCommunicationComplexity,Andros,Greece,pp. 181–196.

[8] Y. Hu, R. Blake and D. Emerson, (1998). An OptimalMigrationAlgorithmforDynamicLoadBalancing,Concurrency:PracticeandExperience10,pp467–483.

[9] Daniel Grosu, Anthony T.Chronopoulos et.al. (2005)‘Noncooperativeloadbalancingindistributedsystems’, JournalofParallelandDistributedComputing,Elsevier,Volume65, Issue9,pp1022–1034.