Randomized Honey Bee Load Balancing Algorithm in Cloud Computing System

Monika Rathore, Sarvesh Rai, Navdeep Saluja

Infinity Management and Engineering College
Sagar (M.P.)

Abstract—This Load balancing is the task of distribution of application tasks to whole all different processors to reduce program execution time. Effective implementation of load balancing can build cloud computing plenty of smart and it jointly improves user satisfaction. Load balancing distributes workloads across multiple computing resources like computers, a laptop pc cluster, network links, central methodology units or disk drives. Load balancing aims to optimize the resource use, maximize the makespan extra on minimize the latency. A load balancing algorithmic program tries to bolster the latency of user’s submitted applications by guaranteeing largest utilization of accessible resources. Load balancing ensures that each one the processors within the system extra as inside the network do around the equal quantity of labour at any instant of your time. To appreciate best machine utilization, tasks from overload virtual machines unit of measurement transferred to the neighbour Virtual machine whose load worth is below threshold. The planned algorithmic rule improves the honey bee forage technique by guaranteeing that no virtual machines keep idle. Once the virtual machine is idle, the algorithmic rule can build multiple tries to steal jobs from a random Virtual machine. This paper, the foremost contribution of Cloud Sim is to produce a holistic code framework for modelling. Whenever some below loaded virtual machines unit of measurement given then no. of tasks do not appear to be to be inflicting to full VMs they are going to be gone to below loaded VMs. For optimize answer and higher latent amount the load ought to be balanced among full and below loaded virtual machines. Throughout this thesis, algorithmic rule is projected named irregular Honey Bee Behaviours based load balancing those targets to appreciate well balanced load across virtual machine.


I. INTRODUCTION

Cloud Computing is that the way forward for technology same by varied executive of IT business. It works on the principle that the user should pay according the time that it's victimisation the resources from varied cloud suppliers. It's customary that it's onerous to handle giant resources. It's really expensive in terms of money equally. so Cloud Computing is that the best answer to those tough problems. Cloud Computing is answer to just about every disadvantage aged by many Industries that area unit directly or indirectly related to the knowledge Technology. This field is growing in no time as several of the massive players among the data Technology field like Microsoft, Google, Amazon, SAP area unit finance uncountable money to induce improved results. They feel that it's the long term of technology

Clusters [1] area unit distributed systems below the superintendence of single body domain. Grid [1] could be a geographically distributed assortment of distributed systems. Cloud could be a assortment of parallel and distributed system wherever the nodes area unit virtualized whereby one physical server will run multiple virtual servers, so reducing the resources also because the price. A cloud is public, personal or hybrid [2]. Personal clouds area unit setup by enterprises for his or her internal use solely. Public clouds are unit setup for public use by the enterprises. The users of a public cloud should comply with the Service Level Agreement (SLA) such that by the cloud supplier. Hybrid cloud could be a combination of personal and public cloud. Open Stack is one the foremost standard computer code accustomed setup a cloud, others being Eucalyptus, OpenNebula, etc. There are unit 3 major sorts of services provided on a cloud: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), computer code as a Service (SaaS)[3].

II. PROBLEM STATEMENT

Until recently the most works on load balancing assumed solid nodes. Usually | this can be | often clearly surrealistic for several instances of Cloud computing, as printed herein, where dynamic and heterogeneous systems are necessary to provide on demand resources or services. New instances of the platform ar started once the load can increase on the way facet predefined thresholds. Therefore, combos of rules impose the circumstances and declare load balancing. as a result of the systems increase in size and quality, these rule sets become unwieldy and it mustn't be potential to require care of a viable observation and response cycle to manage the procedure work. In short, the scale of these systems might exceed the capabilities of connected meta systems to require care of a sufficiently agile and efficiently organized load balancing (or general management) rule set. Load balancing system is required that self regulates the load at intervals the Cloud’s entities whereas not basically having to possess full info of the system. Such self organized regulation might even be delivered through distributed algorithms.

Implementation algorithmic program which might resolve the Virtual machine programming management at a lower place the dynamic atmosphere of the quantity of VMs and requests on Cloud computing, we've projected a flow chart for load balancing in cloud computing environments supported behaviour of honey bee forage strategy. The
tasks are to be send to the under loaded machine and like
forage bee consecutive tasks are also sent there to Virtual
Machine till the machine gets full as flower patches
exploitation is completed by scout bees. Honey bee
behaviour galvanized load balancing, improves the overall
turnout of method and priority based mostly balancing
focuses on reducing the build span , time a task should
facilitate a queue of the VM. Thus, it reduces the response
of sometime of VMs.

III. HONEY BEE BEHAVIOUR IN LOAD BALANCING OF TASKS
Effective implementation of load balancing can build cloud
computing a lot of sensible and it together improves user
satisfaction. inside the planned methodology, a honey bee
forage technique is used for task allocation and loading
levelling. Once tasks area unit assigned to the VMs, current
load is calculated. If the VM becomes full the task is
transferred to the neighbourhood VM whose load price is
below threshold [4]. Honey Bee forage technique employs
decentralised load balancing methodology and task transfer
area unit disbursed on the fly. The algorithmic program
ensures performance of the system and avoid system imbalance.

A) BEE FORAGE BEHAVIOUR
The artificial bee colony formula (ABC) algorithmic rule
supported the intelligent forage behaviour of honey bee
swarm and was planned by Karaboga in 2005 [5]. The
formula is completely galvanized by natural forage
behaviour of honey bees.

B) Initialization technique
Artificial Bee Colony algorithmic rule starts by correlating
all the bees with every which way created food sources.
sure food sources area unit indiscriminately elect by bees
and their nectar amount is ready. These bees come onto the
hive and share the data with bees waiting in dance house [6].
Initialize the population of the scout bees, generate
indiscriminately scout bees into the food sources and
calculate the fitness values.

C) Algorithm
Repeat:

every the utilized bees search round the food
sources and update the new fitness, if the new
fitness is better than the previous values.
choose utilized bees and recruit on looks bees to
go looking round the food sources and calculate on
their fitness worth.
select the onlookers bees with have the most
effective fitness worth.
Send scout bees into the food sources to get new
food sources.
Until (Stopping criterion isn't met)
End At the start, the initial n scout bees
are placed indiscriminately in VM on Cloud
computing and n is that the range of scout bees.

E) EMPLOYED BEE SECTION
Employed bees be the food offer and provide the
neighbourhood of the provision in its memory. once sharing
the information among the dance area, used bees attend
food provide visited by its previous cycle and choose new
food offer by victimization the information among the
neighbourhood. Then spectator prefers a food provide
looking forward to nectar information provided by used bees.

F) ONLOOKER BEE SECTION
Onlooker bees get the information regarding food sources
from used bees in hive and select
one altogether the sources. Spectator bee is anticipating a
dance to determine on a food provide. Waggle/tremble/Vibration dances are performed by the
bees to relinquish a plan regarding quality and quantity of
food and its distance from bee hive.

G) SCOUT BEE SECTION
Scout bee disbursed random search. once the nectar provide
is abandoned by the bees, a current food provide is
indiscriminately determined by a scout bee.

IV. RANDOMIZED HONEY BEE LOAD BALANCING ALGORITHM
In Honey bee behaviour galvanized load balancing
(HBB LB), the simplest way of task allocation and cargo
balancing is place forth. It ignores the idle condition of
virtual machine then finishes up within the wastage of
interval. Therefore the worldwide best answer can't be
detected at intervals a quick span of some time. inside the
planned methodology, accumulated honey bee forage
technique with random stealing is used for task allocation
and load levelling. Once tasks area unit assigned to the
VMs, current load is calculated. If the VM becomes
overloaded the task is transferred to the neighbourhood
Virtual machine whose load value is below threshold.
Honey Bee forage technique employs decentralized load
balancing methodology and task transfer area unit disbursed
on the fly. The formula thus ensures performance of the
system and avoid system imbalance.

Additionally, rudiment consists of three management
parameters:
- Population size (SN) is that vary of food sources
(or solutions) inside the population. Metal is capable the
number of used bees or spectator bees.
- Most Cycle vary (MCN) refers to the foremost
vary of generations.
- Limit is utilized to diversify the search, to examine
the amount of allowable generations that each non-
improved food offer is to be abandoned.

Let\(=\{VM1, VM2, VM3… VMN\}\) is a set of N virtual
machines and Task\(=\{task1,2, task3, … ,K\}\) of K task to be
regular and processed in VM. All the machines are
unrelated however are paralleled. Programming is non-
 preemptive which suggests that the process of the tasks on
VMs can’t be interrupted.

Algorithm
1. Get the available virtual resources from data
center.ie, VM1, VM2,… VMn, list of tasks T=T1, T2,…Tn by the user.
2. When a request comes, the scheduler finds the
Expected computing capacity for tasks
3. Compute the average computing capacity for each
task using the equation,
4  Find the load of VM  
5  Compute the average system load  
6  Compute Load  
   The probability value is checked for confinement within the range 0 to 1 as,  
   i.  \( 0 < P < 1 \)  
      1.  \( \text{Underloaded list} = \text{VM}_i \)  
   ii.  else  
      1.  \( \text{Overloaded list} = \text{VM}_i \)  
7  Select Under loaded VMs and compare its Average computing capacity with Expected computing power of tasks.  
   Check if \( \text{ACAP} \leq \text{ECAP} \), then VMs are marked as Fittest and tasks are allocated to it  
8  After task allocation to VMs, some VMs remains underutilized. This leads to wastage of processor time  
   Check  
   If \( \text{system load} \leq \text{TRS_LOW} \) Perform Randomization  
   Select VMs with \( \text{system load} > \text{TRS_HIGH} \)  
   Randomly Select jobs from those VMs and allocate to VMs with system load \( \leq \text{TRS_LOW} \)  
   If there are \( N \) VMs, the algorithm will make \( N-1 / N \) attempts on an average to steal a job.

The probability value is checked for confinement within the range 0 to 1 as,  
1. If \( 0 < P < 1 \)  
   1. Underloaded list = VM\(_i\)  
2. else  
   1. Overloaded list = VM\(_i\)

After task allocation to VMs, some VMs remains underutilized. This leads to wastage of processor time  
Check  
If \( \text{system load} \leq \text{TRS_LOW} \) Perform Randomization  
Select VMs with \( \text{system load} > \text{TRS_HIGH} \)  
Randomly Select jobs from those VMs and allocate to VMs with system load \( \leq \text{TRS_LOW} \)  
If there are \( N \) VMs, the algorithm will make \( N-1 / N \) attempts on an average to steal a job.

**Fig: 1. Flow chart of VM programming and cargo balancing victimization rudiment.**

**V. EXPERIMENTAL RESULTS**
According to the algorithmic rule pictured quite, the simulation victimization CloudSim-3.0.3 Tools unit of measurement addressed. There unit of measurement four servers unit used here. The parameter setting of rudiment algorithmic rule is as follows. The experiment is shown at intervals the graph that consists of comparison of resource usage to the servers. The bar diagram shows the entire memory and then the used memory. The comparison of build span for load balancing pattern honey bee galvanized load reconciliation algorithmic rule (HBB LB) and randomised honey bee algorithmic rule with organization (HBB RS) is illustrated in Fig.1. Makespan could also be made public as a result of the general task completion time. We have a tendency to tend to denote completion time of task Ti on VMj as C\(_{ij}\).  
Make span = \( \max \{C_{T_i} | 1 \leq i \leq n; 1 \leq j \leq m\} \)

The axis represents the amount of tasks and axis represents Makespan in milliseconds. With load balancing pattern honey bee galvanized load balancing (HBB LB), the makespan is reduced considerably. Once vary of tasks can increase, the excellence in create span is extra and increased honey bee rule with randomised is further economical. The comparison is in addition created in terms of latency. Fig.2 shows the latency in milliseconds for HBB LB and HBB R and. The axis represents vary of tasks and axis represents the latency in milliseconds. It is the range of sometime taken between submission of asking and thus the first response that is created. The reduction in waiting time is beneficial in up the responsiveness of the VMs. From this graph, it’s clear that accumulated honey bee hunt technique with randomisation is a lot of sensible and provides higher performance.

**Fig: 2. of Make span for Load balancing mistreatment Honey Bee rule**

The axis represents the amount of tasks and axis represents Makespan in milliseconds. With load balancing pattern honey bee galvanized load balancing (HBB LB), the makespan is reduced considerably. Once vary of tasks can increase, the excellence in create span is extra and increased honey bee rule with randomised is further economical. The comparison is in addition created in terms of latency. Fig.2 shows the latency in milliseconds for HBB LB and HBB R and. The axis represents vary of tasks and axis represents the latency in milliseconds. It is the range of sometime taken between submission of asking and thus the first response that is created. The reduction in waiting time is beneficial in up the responsiveness of the VMs. From this graph, it’s clear that accumulated honey bee hunt technique with randomisation is a lot of sensible and provides higher performance.

**Fig: 3. Response Time**
Fig. 3 illustrates the degree of imbalance in load in terms of range of tasks. The coordinate axis represents the quantity of tasks and coordinate axis represents the imbalance degree. Imbalance degree is outlined in equation give below,

\[ \text{Degree of imbalance} = \frac{\text{T}_{\text{high}} - \text{T}_{\text{low}}}{\text{T}_{\text{avg}}} \]

Where \( \text{T}_{\text{high}} \) is that the best task, \( \text{T}_{\text{low}} \) is the lowest task among all the virtual machines and \( \text{T}_{\text{avg}} \) is that the common task of virtual machines. From Fig. 5.5, it’s clear that imbalance degree could be a smaller quantity for enlarged honey bee galvanized load reconciliation of tasks with organization and so its performance is further compared to ancient honey bee hunt load balancing technique.

\[ \text{Resource Utilization} = \frac{\text{VM demand}}{\text{range of tasks}} \]

Fig. 4 shows the resource utilization rate of projected technique with random stealing that is comparatively high compared to existing honey bee load balancing technique. With randomization technique, tasks area unit purloined from a random Virtual machine once a VM is idle. It therefore saves the idle time of the process parts within the Virtual machine.

VI. CONCLUSION

Rudiment algorithm is suitable for the cloud computing atmosphere as a result of the formula is in a position to successfully build use of the exaggerated system resources and reduce makespan. This application has experimentally proves that the planned algorithm for cloud load management supported particle randomisation algorithm scale back issues and improvement with random algorithms. the foremost goal of the plan need algorithm is minimizing the complete execution time of giving tasks. The algorithms in applications with the number of tasks varied from fifty to k evaluated. The experimental results illustrated that the planned ways in which of rudiment performed effective results than all ways in which and its performance could be a ton of outstanding in quantify ability. It together accustomed inhibit the system crash. It aims at achieving the upper programming conservation throughout the semi permanent operation of a cloud information centre, and protects the performance of a VM running. For any studies, the preventative Virtual machine hardware is accessed by multiple user at constant time area unit focused.

VII. FUTURE WORK

We have an inclination to rearrange to enhance this algorithm by considering different QoS factors of tasks. The performance of the given algorithms will even be increased by variable entirely different parameters. Projected algorithm remains a promising and interesting algorithm, which could still be extensively utilized by researchers across varied fields. Its potential advantage of being merely hybridized with whole completely different meta-heuristic algorithms and components makes it robustly viable for continued utilization for additional exploration and improvement prospects in additional years to come back.
REFERENCES


