A Method Towards Allocation of VM using a Cuckoo Search Algorithm

Roma Singh, Rohit Bajaj, Jaspreet Singh

Abstract*: *Cloud computing is a new technology created for complex systems with massive service sharing. It is not easy to analyze the cloud model due to its complex and large scale structure. To meet the user's requirement along with the quality of service (QoS) parameters become the most dominating research issue. The QoS parameters are affected due to various issues such as load balancing, Virtual Machine (VM), makespan, throughput, Utilization of Hosts' Processing Capability (UHPC) and many more. The allocation policy of VM mainly used to place VMs on hosts at various datacenters. This article outlines the new VM allocation policy using Modified Best Fit Decreasing (MBFD) along with with Cuckoo Search (CS) algorithm that delivers the VM burden between hosts to improve the handling capability and the cloud system performance. The test results are obtained using CloudSim trace simulation and the QoS are compared with existing allocation policies of VM.

Keywords*: *Cloud computing, MBFD, Simple policy, Round Robin, CS, Throughput, Makespan, UHPC, VM and Host.

I. INTRODUCTION

Cloud computing is widespread, largely used due to virtualization requirement, utility calculation and service-oriented structural design [1]. The genral idea of cloud computing is to provide services to the cloud users or customer based on demand. Cloud computing has offered services like Software as a service (SaaS), infrastructure as a service (IaaS), and Platform as a service (Paas) to their customer [2]. In addition, it provides users with flexibility, scalability and the to deliver the best performance in line with the application's requirements. It provides cloud user with large data which is accessible by the user later on can be accessed from anywhere [3-5]. Cloud computing is driven by scale economics. Cloud system uses virtualization technology to provide users with cloud resources (e.g CPU, memory) in the virtual machine [6].

The user adheres to the learning environment to offer computational resources for the virtual machine, and research needs of a sandbox for the application. According to the source vendor point of view, the operating costs that comprise the large-size data center system e-account must be minimized [7]. Therefore the allocation of VM in virtualized data centers becomes a challenging task and is appear on a static and dynamic map. A VM allocation is employed to describe each VM to PMs to optimize a specific objective feature. The goal function is to maximize productivity, minimize power / or maximize the profitability of the provider [8]. In this article, a new VM allocation policy by using the concept of optimization schme has been applied to a VM from a designated vm_list to a VM with a load balancing method for allocating a VM to a host with a maximum load capability among all hosts host_list [9]. The main purpose of the proposed allocation policy is to separate the VM with maximum processing capability. Moreover, separation policy distributes load among all hosts based on the residual load [10]. This suggested policy improves the handling capabilities of hosts and reduces the impact on the timing of cloudlets. Thus, the development of the cloud system has been improved [11].

A. *VM Allocation Policies*

VM allocation policy helps users to create a VM instance on a host within a defined datacenter. Proper deployment of VMs helps to handle the processing capabilities in a well defined way. Also, it can help to lessen the end time along with make span [12]. This leads to better transmission of the cloud system. A good VM allocation policy is a challenge in cloud computing. In this research, the Cuckoo search is used as an optimization algorithm to optimize the properties of the host. To show the efficiency of the proposed work, a comparison between different existing techniques used by various researchers have been studied and provided [13].

- Modified best fit (MBFD)

In the proposed work, an energy optimization model is used which is based on the concept of VM migration in cloud computing. During job scheduling, VMs are allocated to the host machine by using the MBFD algorithm. MBFD algorithm is used to migrate the Virtual machines to a particular physical machine in case of overloaded, under loaded and normal loaded with low energy consumption [14].

- Simple VM allocation policy with Round Robin VM Allocation

Sourav Banerjee et al. (2017) have employed a simple VM allocation policy which is inherent in Cloudsim tool, that works in the same fashion as that of FCFS allocation. This algorithm selects the host that used less number of PMs and hence allocated the VM by crosschecking the hardware requirement.

Additionally, in round robin allocation scheme, VMs are formed under FCFS, but the host selection procedure differs slightly from the simple allocation policy.

```
Algorithm : Modified Best Fit Decreasing (MBFD)
Input: host_list, Vm list Output: VMs Allocation
Vm_list.sort decreasing utilization
( )
```

Revised Manuscript Received on July 10, 2019.
Roma Singh, Chandigarh University, Mohali, India
Rohit Bajaj, Chandigarh University, Mohali, India
Jaspreet Singh, Chandigarh University, Mohali, India

```
For each Vm in Vm list do
    minPower← $max$
    aloocated host← $null$
    for each host in host list do
        if host has enough resource for Vm then
            $power \leftarrow estimate\ power\ (host, Vm)$
                If power<manpower then
                    Allocate host← $host$
                    Mini power← $power$
  if Allocated host≠ $null$ then
    Allocate VM to allocated host
Return Allocation
```

```
Input: host _list, Vm_list
Output: Allocated VM to host
1. Initialize all the VM allocation state to be available in the
   VM list as per the processing ability do
2. For each $host_i$ in the host_ list do
3. Determine remaining load capacity (RLC) of host;
4. End
5. Assign $host_i$= max $host_{RLC}$;
6. If allocation unsuccessful then
7. Host_list=host_list-$host_i$;
8. End
9. If VM is not assign to any host then
10. The assignment of $VM_i$ is unsuccessful;
11. End
12. Else
13. $VM_i$ is assigned to $host_i$
14. End
15. End
```

In this algorithm both simple and round robin is combine and allocate VM with maximum MIPS to the host along with the maximum remaining load capacity. This scheme was used by Sourav Banerjee et al. (2017) as an allocation policy that provides better results in terms of the host's processing capability. The algorithm for a round robin with the simple policy is written above.

- Firefly Algorithm (FA)

Nidhi Jain Kansal and Inderveer Chana (2016) have used firefly as an optimization algorithm in their research. This optimization approach will reduce the energy consumption, nodes and quantities of cloud data centers by reducing the amount of data stored on the green computer. In their research FA is used to migrate the maximum loaded VM's to the least loaded VM by maintaining the energy of the datacenter. The general description is provided below. The Firefly is a metaheuristic algorithm inspired by the lightning behaviors of flies. The main objective of flies is to attract other flies by behaving like as a single system.

```
Algorithm 3: Firefly Algorithm
Input: Group of jobs, set of VMs, resource utilization &
energy consumption
Output: Determine the best VM
Begin
VM()
For (each VM) do
Calculate energy consumption
EC→ energy consumption
Determine Execution time
ET→ Execution time
For (each VM) do
Determine attraction index
AI→ Attraction index
Arrange AI in rising order as per the energy consumption
value
Find the Vm that consume less energy as per the distance
between the nodes
For (each VM on the starting place node) do
Calculate load
Arrange load in descending order
Assign sorted job to the destination PM
```

- First come first serve

Calheiros et al. (2009) have utilized First come first serve (FCFS) algorithm. In this approach, the tasks that arrive first are executed first by the processor. The tasks are added into the queue, and the task, which enters first leave first from the queue. The proposed algorithm is described below:

```
Algorithm: FCFS
Input: Initialize VM and maintain the table of VM servers i.e.
VM busy and VM available
Output: Allocated VM Table
Initialize parameters
t1 = 0                        // Total time
t2 = 0                        // Turnaround time
n = Host                      // no of hosts
btime = []                    // burst time
wtime = zeros(1,n)            // waiting time
tatime=zeros(1,n)             // turnaround time
For i=2→n
  wtime(i) = btime(i-1)+wtime(i-1)  // waiting time will be
sum of burst time of previous process and waiting time of
previous process
  t1=t1+wtime (i)                // calculating total time
End
For i=1→n
   tatime(i)=btime(i)+wtime(i)     // turnaround time=burst
time +wait time
   t2=t2+tatime (i)                 // total turnaround time
End
For i=1→n
   Allocated VM Table = Host according to above conditions
End
Return: Allocated VM Table
End
```

B. *Round Robin Policy*

One of the simplest planning techniques that use the timeframe principle is known as RR. The time is divided into more than one slice and each node is given a certain time slot or time interval, which means that it uses the principle of time schedules. For every node a quantum and its operation has been given. The service provider's resources are provided to the customer who requests the time zone.

```
Algorithm: RR
Input: VM List and Host
Output: Allocated VM Table
Initialize Variables
n=Host          // no of hosts
btime=[]        // burst time

q=2             //quantum time
tatime = zeros(1,n)    // turnaround time
wtime = zeros(1,n)   // waiting time
rtime = btime          // initially remaining time is equal to waiting time
b=0
t=0
Flag =0            // this is set if process has burst time left after quantum time is completed
For i=1→n          // running the processes for 1 quantum
    If (rtime(i)>=q)
        For j=1→n
            If (j==i)
                rtime(i)=rtime(i)-q          //setting the remaining time if it is the process scheduled
            Else if (rtime(j)>0)
                wtime(j)=wtime(j)+q    // incrementing wait time if it is not the process scheduled
            End
        End
    End
    Else if (rtime(i)>0)
            For j=1→n
                If (j==i)
                rtime(i)=0            // as the remaining time is less than quantum it will run the process and end it
                Else if (rtime(j)>0)
                    wtime(j)=wtime(j)+rtime(i) // incrementing wait time if it is not the process scheduled
                End
            End
        End
    End
  End
End
For i=1→n
    If (rtime(i)>0)     // if remaining time is left set flag
        Flag = 1
    End
End
While (Flag==1)         // if flag is set run the above process again
   Repeat Above process
  Allocated VM Table = Host according to the above conditions
End
Return: Allocated VM Table
End
```

C. *Memory Accesses Per Cycle (MPC) Balance Algorithm*

This algorithm is used to balance the energy as well as the performance of the server across the cluster. This algorithm initially checks the nMPC value. The case, when the nMPC is less, then the function returns back by indicating that no balancing is needed for task n1. The case, when the value of nMPC is more, then determines the VM with minimum vMPC in n1. This can be migrated to a PM with minimum nMPC than n1 and obtained a better MPC balance in the server. The PM target is saved into the PM-min and initialized NULL. The algorithm is written from step 6 to 12 the algorithm tries to find the destination PM with minimum n MPC, with the condition mentioned in the algorithm. The case when the condition such as nMPC is find minimum, the value is stored into the PM-min. In this manner, the algorithm is completed and allowed the selected VM to the defined PM with proper load balancing. The algorithm is written below:

```
Algorithm: MPC
Input: PM n1
Output: Migrated VM
if $nMPC_{n1} < nMPC_{th}$ then
 return
end if
$vm_{min} \leftarrow$ min_mpc_vm(n1)
 pm_min ← NULL
 for all PMs $n_i$ except $n_1$ do
if ($nMPC_{ni} < nMPC_{th}$) and ($nMPC_{n1} < -nMPC_{th}$) > ($nMPC_{th} - nMPC_{ni}$) then
 if !pm_min or $nMPC_{pm\_min} > nMPC_{n1}$ then
 $pm_{min} \leftarrow$ ni
 end if
end if
 end for
if $pm_{min}$ and ($nMPC_{n1} - nMPC_{pm\_min}$) > $uMPC_{vm\ min}$ then
 do_migrate($vm_{min}$, n1, $pm_{min}$)
 end if
```

II. RELATED WORK

The work done by various researchers described above is shown in the tabular form as written below:

III. PROPOSED WORK

In this paper, we designed a new VM allocation scheme along with a suitable load balancing method. In the proposed work, we assume that there is no load and after applying tasks the RLC is calculated by using the formula written below.

$$RLC = 100 - \left(\frac{L_{ph}}{AL_{maxh}}\right) \times 100\%$$

Here L_{ph} represents the present load on host

AL_{maxh} Signifies the allowable load on host

Table 1. Comparative Analysis Of Existing Techniques For Vm Allocation In Cloud Computing

Researchers	Aim	Algorithms	Advantages	Limitations

Sourav Banerjee et al. (15, 2017)	Novel VM allocation policy has been designed to provide better VM allocation	VM allocation simple and Round Robin VM allocation	Allocating VM using defined techniques mentioned under algorithm column lessen the makespan of cloudlet along with the improved throughput of the cloud system	The concept of energy consumption have not been considered
Nidhi Jain Kansal and Inderveer Chana (16, 2016)	an energy-aware VM migration has been done using an optimized algorithm for cloud computing	Firefly algorithm	The average energy consumption has increased. faster convergence speed The attraction properties of FF algorithm helps to reduce the energy consumption problem.	This technique works only to migrated overloaded VMs.
Gaurav Dhiman et al. (17, 2009)	V Green technique using MPC as a load balancing algorithm has been presented to optimize the energy in a virtualization environment.	Memory Accesses Per Cycle (MPC) Balance Algorithm has been used	The energy has saved up to the desired level. The algorithm works with higher efficiency in case of varying load `	Complexity is high
Anton Beloglazov et al. (18, 2012)	This work enhances cloud computing in two ways. Initially, the authors have contributed to save energy of datacenter and also help in the reduction of CO2 emission.	Modified Best Fit Decreasing (MBFD) and Minimization of Migrations (MM)	The proposed approach tends to considerable reduction of energy consumption in Cloud data centres than that of static resource allocation scheme	Loading effect has not been considered
Calheiros et al. (20, 2009	Clouds system has been developed to execute the cloudlets.	First, come first server	It can handle complexities with higher accuracy	Authors have mainly focused on schedule length and reduction of energy consumption

CS was inspired by the guaranteed parasitism of some bird species, putting eggs in the nests and go for searching food. After the searching process of food, these birds came back to their host and check eggs lay by them. If the eggs are broken by natural process and produce bird then it is ok, otherwise; the bird breaks the eggs by itself or leaves the nest. Similar work is performed by CS algorithm, in this research. Here, the RLC value is assigned to the egg value and the allocation of VM has been adjusted as per the fitness function. This approach helps to enhance the performance of the datacenter.

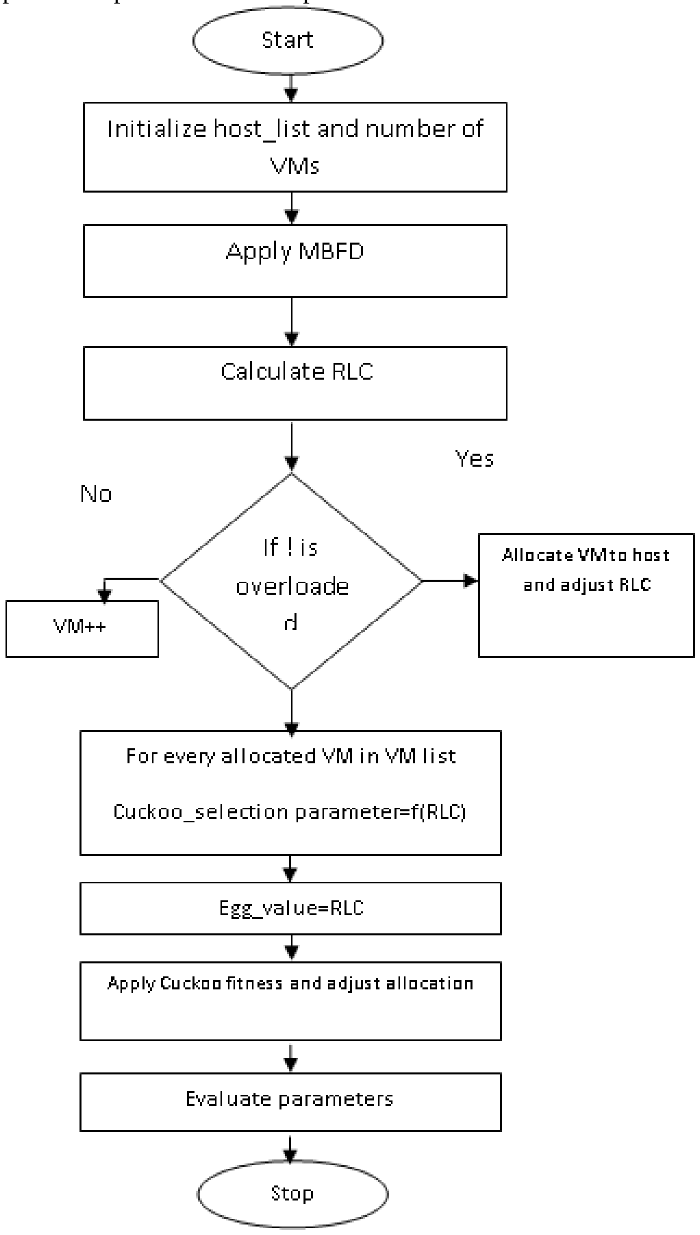
Start
Initialize host_list and number of VMs
Apply MBFD
Calculate RLC
Yes
No
If ! is overloaded
Allocate VM to host and adjust RLC
VM++
For every allocated VM in VM list
Cuckoo_selection parameter=f(RLC)
Egg_value=RLC
Apply Cuckoo fitness and adjust allocation
Evaluate parameters
Stop

Fig. 1 Flow of proposed work

Initially, number of VM's (6) and number of hosts (2) denoted by host 0 and host 1 are initiated. MBFD algorithm is applied to short the VM; the RLC of shorted VMs is calculated by using the formula in equation (1). In case if the VM's are overloaded then adjust RLC, otherwise; increment VM by one. Add VM in the VM list table and select fitness function by applying CS algorithm.

Assign RLC value to the egg value of the CS algorithm. CS algorithm adjusts the RLC and enhances the performance. At last calculate and compare the parameters.

The proposed algorithm is

written below.

```
Algorithm: Proposed algorithm
Input: host _list, Vm_list
Output: Allocated VM to host
Initialize all the VM allocation state to be available in the VM list as per the processing ability do
For each $host_i$ in the host_ list do
Determine remaining load capacity (RLC) of host;
End
Assign $host_i$= max $host_{RLC}$;
If allocation unsuccessful then
Host_list=host_list-$host_i$;
End
If VM is not assign to any host then
The assignment of $VM_i$ is unsuccessful;
Determine fitness and rank eggs;
While (i>max generation)
VM=i++
Best cuckoo carry out levy flights and lay eggs ($y_{new}$) in all nests;
Compute fitness, $y_{new}$;
If ($y_{new} < y_x$)
Allocate VM’s to under loaded PM
End if
Novel eggs ($y_{new}$) are laid by host birds via Levy flights along with a mutation function of $M_u$.
Evaluate quality/ fitness, $y_{new}$);
If ($y_{new} < y_x$)
Replace present solution with novel solution
End if
Allocate VMs to an appropriate PM
End while
End
```

IV. RESULT ANALYSIS

A. Dataset used

The entire process of the work is shown in figure 1. In this article, we have considered two hosts, 6 numbers of VMs with 12 cloudlets. Each host is denoted by a unique identity (host 0 and host 1) along with their processing capability. The entire processing capability of host is determined by using the formula written below.

$$TC_{host} = Pc_{host} \times PC_{pre}$$

TC_{host}- Host’s total processing capability

Pc_{host} Represents the number of processing elements present in a host

PC_{pre} Signifies the processing capability of a processing element

The performance of the designed system has been analyzed on the basis of different computational parameters such as RLC, makespan, Throughput and UHPC as shown in the figure below.

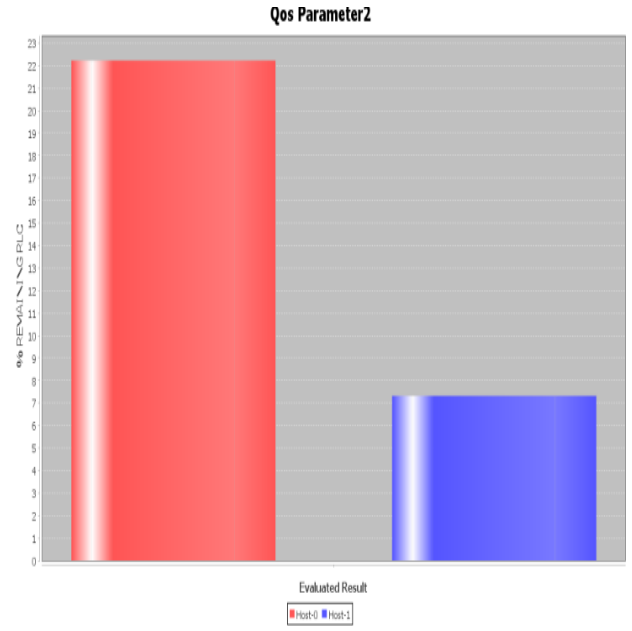

Fig. 2 Remaining load capacity

Fig. 2 represents the RLC in the case when there is no host and when the host is allocated to the system. From the figure it is clear that with host 0 the RLC is less as compared to host 1.

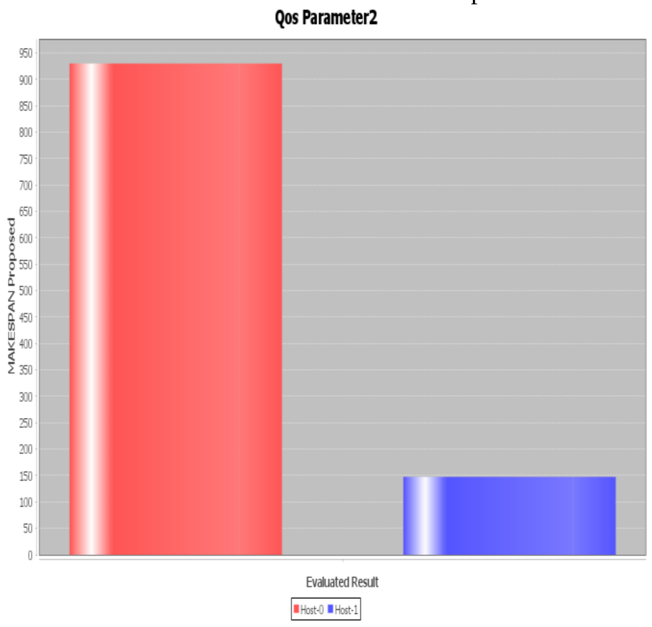

Fig. 3 Makespan of Proposed

Makespan represents the time between the initialization of VM to the allocation of VM. The makespan measured for host0 and host 1 for the proposed work is 1400 and 220 respectively. The average value of makespan for the proposed work is about 830 has been determined. 1.21 % of reduction from the base work has been determined. In the figure above as the cuckoo search is applied in the proposed system the makespan decreases. This is due to the fact that cuckoo search classifies the overloaded and underloaded hosts, therefore, allocate the VM’s to the underloaded host so that load can be balanced and makespan get reduced. The makespan is calculated by using the formula written below.

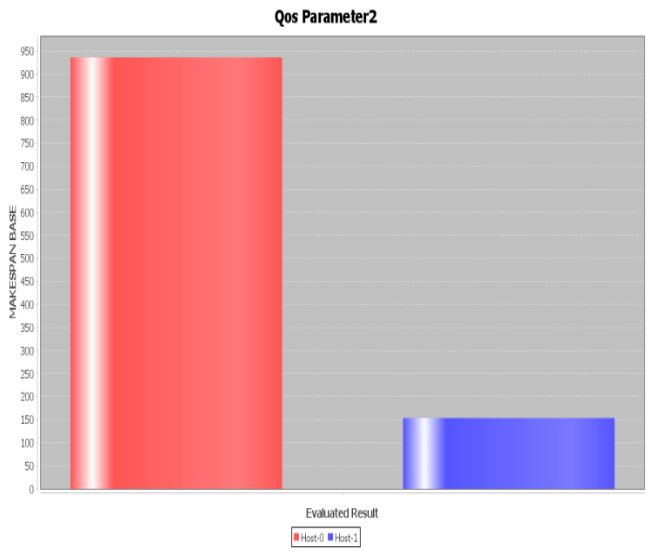

Fig. 4 Makespan of Sourav Banerjee et al. (2017)

The makespan measured for host0 and host 1 for the proposed work is 940 and 150 respectively. The average value of makespan for the proposed work is about 545 has been determined.

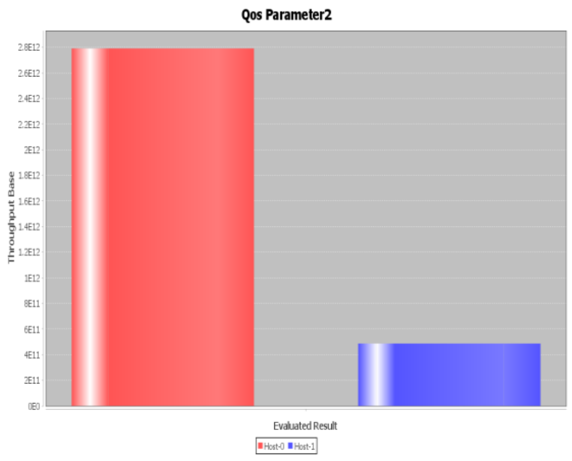

Fig. 5 Throughput of Sourav Banerjee et al. (2017)

The throughput calculated by utilizing the algorithm proposed by Sourav Banerjee et al. (2017) i.e. the combination of simple policy and round-robin policy for host 0 and host 1 is 2.81 and 0.611 respectively. The average value of about 1.71 has been obtained.

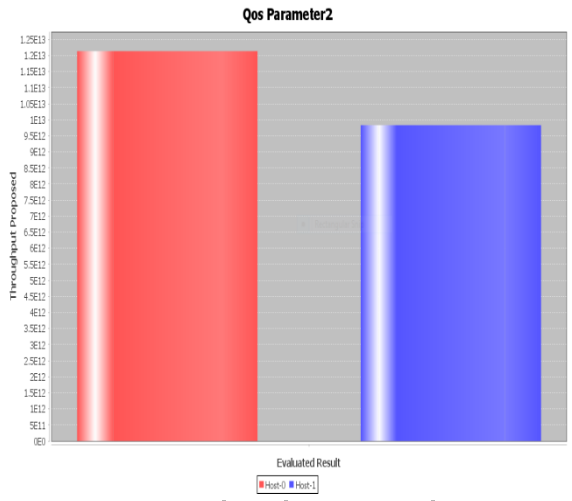

Fig. 6 Throughput Proposed

By balancing load using an optimization algorithm named as CS the throughput value has been increased for both host 0 and host 1, the increased values are 1.213 and 0.951 respectively. The average value of about 1.82 has been obtained. Thus there is an enhancement of about 6.43 % has been obtained.

Throughput = Total number of Processed Cloudlets /makespan

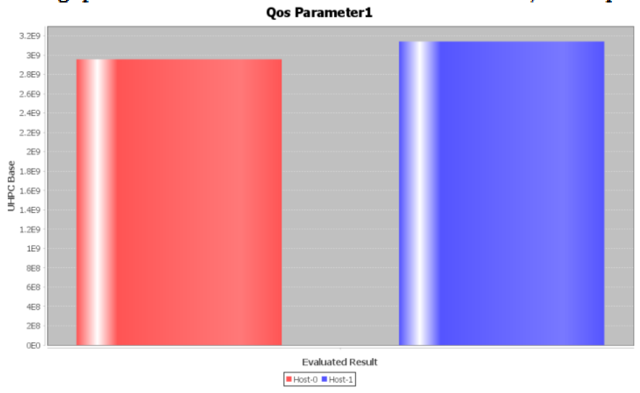

Fig. 7 UPHC of Sourav Banerjee et al. (2017)

It represents the total processing capability of VMs, the values of UPHC examined for Sourav Banerjee et al. (2017) for host 0 and host 1 is 3.04 and 3.27 respectively.

UHPC=$(PC_{AL_{VM}}/PC_{host}) \times 100$

$(PC_{AL_{VM}}$ denotes the sum of all processing capabilities of allocated VMs

PC_{host} - Signifies the entire processing capability of host

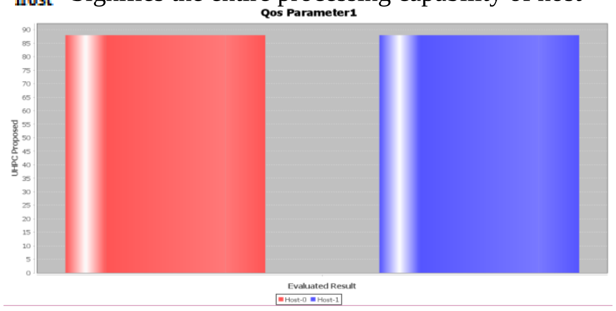

Fig. 8 UPHC of Proposed

The total UPHC measured for the proposed work with host 0 and host 1 is 87 and 86 respectively. The average value of UPHC measured for the proposed work is 86.5 respectively. Therefore there is an enhancement of 10.46 % in the processing capability of VM has been observed.

V. CONCLUSION

This paper highlights the new VM allocation policy by using the concept of optimization scheme that provides better UHPC. We have also demonstrated that using our allocation policy along with the concept of load balancing has resulted in the better allocation of VM, the reduction of the makespan, as well as enhances the capacity of the cloud system. Also, the computing parameters have been improved in terms of throughput, makespan and UHPC. The percentage increase in the parameters such as throughput, makespan and UHPC is about 6.43%, 1.12 % and 10.46 % respectively.

In future the the work can be

extended by increasing VM's, PM's as well as cloudlets.

REFERENCES

1. B. Yang, F. Tan, Y. S. Dai, & S. Guo, "Performance evaluation of cloud service considering fault recovery,"In IEEE International Conference on Cloud Computing (pp. 571-576). Springer, Berlin, Heidelberg, 2009, December.
2. N. Tziritas, T. Loukopoulos, S. Khan, C. Z. Xu, & A. Zomaya, "A Communication-Aware Energy-Efficient Graph-Coloring Algorithm for VM Placement in Clouds," In 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation(SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI) (pp. 1684-1691). IEEE, 2018, October.
3. M. S. A. Latiff, S. H. H. Madni, & M. Abdullahi, " Fault tolerance aware scheduling technique for cloud computing environment using dynamic clustering algorithm," Neural Computing and Applications, Vol.29, no.1,pp. 279-293, 2018.
4. E. Jonas, J. Schleier-Smith, V. Sreekanti, C. C. Tsai, A. Khandelwal, Pu, Q., "Cloud Programming Simplified: A Berkeley View on Serverless Computing," arXiv preprint arXiv:1902.03383, 2019.
5. S. Svorobej, P. Takako Endo, M. Bendechache, C. Filelis-Papadopoulos, K. M. Giannoutakis, G. A. Gravvanis,, "Simulating Fog and Edge Computing Scenarios: An Overview and Research Challenges," Future Internet, Vol.11, no.3, pp.55- 67, 2019.
6. S. Dam, G. Mandal, K. Dasgupta, & P. Dutta, "An ant-colony-based meta-heuristic approach for load balancing in cloud computing,"In Applied Computational Intelligence and Soft Computing in Engineering (pp. 204-232). IGI Global, 2018.
7. T. Chatterjee, V. K. Ojha, M. Adhikari, S. Banerjee, U. Biswas,, & V. Snášel, "Design and implementation of an improved datacenter broker policy to improve the QoS of a cloud," In Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing and Applications IBICA 2014 (pp. 281-290). Springer, Cham, 2014.
8. Y. Feng, W. Zhijian, X. Feng, Z. Yuanchao, Z. Fachao, & Y. Shaosong, "A novel cloud load balancing mechanism in premise of ensuring QoS.,"Intelligent automation & soft computing, Vol.19, No.2, pp.151-163, 2013.
9. N. Pasha, A. Agarwal, & R. Rastogi,, "Round robin approach for VM load balancing algorithm in cloud computing environment," International Journal of Advanced Research in Computer Science and Software Engineering, Vol.4, no.5, pp.34-39, 2014
10. D. G. Amalarethinam, & F. K. M. Selvi, "A minimum makespan grid workflow scheduling algorithm. In 2012 International Conference on Computer Communication and Informatics (pp. 1-6). IEEE, 2012, January.
11. Y., Zhijian Feng, W., Feng, X., Yuanchao, Z., Fachao, Z., & Shaosong, Y, "A novel cloud load balancing mechanism in premise of ensuring QoS. Intelligent automation & soft computing, Vol. 19, No.2, pp.151-163, 2013.
12. B. Wickremasinghe, R. N. Calheiros,, & R. Buyya, "Cloudanalyst: A cloudsim-based visual modeller for analysing cloud computing environments and applications. In 2010 24th IEEE international conference on advanced information networking and applications (pp. 446-452). IEEE, , 2010, April.
13. K. Wee, R. Mardeni, S. W Tan, & S. W. Lee, "QoS prominent bandwidth control design for real-time traffic in IEEE 802.16 e broadband wireless access," Arabian Journal for Science and Engineering, Vol.39, No.4,pp. 2831-2842, 2014.
14. S. Banerjee, M. Adhikari, S. Kar, & U. Biswas, "Development and analysis of a new cloudlet allocation strategy for QoS improvement in cloud," Arabian Journal for Science and Engineering, Vol.40, No.5, pp.1409-1425, 2015.
15. S. Banerjee, R. Mandal, & U. Biswas, "An Approach Towards Amelioration of an Efficient VM Allocation Policy in Cloud Computing Domain," Wireless Personal Communications, Vol. 9, No.2, pp.1799-1820, 2018.
16. N. J. Kansal, & I. Chana, " Energy-aware virtual machine migration for cloud computing-a firefly optimization approach," Journal of Grid Computing, Vol.14, No.2, pp.327-345, 2016.
17. G. Dhiman, G. Marchetti, & T. Rosing, "vGreen: a system for energy efficient computing in virtualized environments," In Proceedings of the 2009 ACM/IEEE international symposium on Low power electronics and design(pp. 243-248). ACM, 2009, August.
18. A. Beloglazov, J. Abawajy, & R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," Future generation computer systems, Vol. 28, No.5, pp.755-768, , 2012.
19. R. N. Calheiros, R. Ranjan, C. A. De Rose, & R. Buyya, "Cloudsim: A novel framework for modeling and simulation of cloud computing infrastructures and services," arXiv preprint arXiv:0903.2525, 2009.

AUTHORS PROFILE

Roma Singh is a full-time research scholar at the Computer Science Engineering Department, Chandigarh University, Punjab, India, 2017-19. She has completed her graduation in IT from Jaypee University of Information Technology, Waknaghat, Solan,H.P. Her area of interest for research are big data and cloud computing and has done project on IOT in B Tech.

Dr. Rohit Bajaj received his B-Tech degree from GJUS&T, Hisar and M-Tech degree from MDU, Rohtak . He received Ph.D. degree from Sai Nath University, Ranchi. Presently, he is an Associate Professor in the Department of Computer Science & Engineering, Chandigarh University, Gharuan Mohali, Punjab, India. He has 10 years of experience of Teaching and Research as well as guiding Post Graduate & Ph.D. Students. He has several papers in referred journals of national and international repute to his credit. His research area of interest is Data Science,Cloud Computing, Wireless Sensor Networks, Soft Computing.

Dr. Jaspreet Singh is working with Department of Computer Science and Engineering, Chandigarh University, Gharuan, Mohali,(Punjab) India. He received M.Tech degree in Computer Science and Engineering from Punjab Technical University, Jalandhar, Punjab, India and obtained Ph.D. degree from Department of Computer Science and Engineering, Maharishi Markandeshwar University, Ambala,(Haryana),India. He also has published several research papers in national as well as international Journals and conferences of repute. He has research contributions in the area of cloud computing, networking, security and cryptography and image processing.