International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 24 (2017) pp. 15782-15790

Energy and Cost Efficient Dynamic Load Balancing Mechanism for Resource Provisioning in Cloud Computing

Shreenath Acharya
Asst. Professor, Department of Computer Science & Engineering
St. Joseph Engineering College, Vamanjoor, Mangaluru, Karnataka, India.
Orcid: 0000-0003-3834-0003

Dr. Demian Antony D'Mello
Professor & HOD, Department of Computer Science & Engineering
Canara Engineering College, Benjanapadavu, Mangaluru, Karnataka, India.

Abstract

Cloud computing paradigm has gained enormous success through its heterogeneous services and diverse utility. The widespread access of cloud technology requires due considerations to some of the vital parameters about its available resources under service. Among them response time, load balancing, energy consumption and resource utilization are of primary concern. In this paper, a dynamic load balancing policy based on the performance, resource utilization and the power consumption is implemented. It checks for the performance and the load factors in order to allocate a resource for users' jobs request. It also considers resource utilization and power consumption as a factor based on it. The results obtained after the simulations with various combinations of the datacenters clearly indicate that the proposed method results in effective reduction in response time and power consumption and improved resource utilization by comparing with the considered existing mechanisms.

Keywords: Cloud Computing, Load Balancing, Resource Utilization, Game Theory, Fuzzy Logic, Bayesian Learning, Artificial Neural Network

INTRODUCTION

The development of technology has resulted in profound utilization of cloud computing in the 21st century as a model for on-demand internet based computing. This is mainly due to its support for features like scalability, multi-tenancy and parallel computations using the virtualization technique. Among all the benefits cloud offers there is always involvement of complexities and the threats during the provisioning of the virtual machines (resource). The varied forms of services offered by cloud would pave the way for hidden complexities which need to be handled appropriately by the service providers.

The factors generally considered during cloud provisioning are makespan, response time, power/energy consumption, cost, resource utilization and the load balancing. Among these we have focused more on the response time, power consumption and resource utilization by way of load balancing. Load balancing is a phrase for workload distributions across multiple computing resources [1] such as servers, a cluster, networks, disk drives etc. It seeks to acquire resource usage optimization, increased throughput and improved performance along with eliminating the possibilities of overburden of the resources. Load balancing applied to multiple components would result in enhanced reliability through redundancy.

Load balancing could be performed statically or dynamically. Static load balancers does not use current state of the system [2] and are not pre-emptive, hence are less preferred, but dynamic load balancers use current state of the system and permit pre-emption by allowing the processes to move from over-utilized to under-utilized systems. It could be handled at sever level or the virtual machine level. We have considered dynamically balancing the load at the server level for our experimentation purposes.

The main problem with load balancing in cloud computing lies with resource utilization and thereby power consumption and performance. Most of the loads balancing strategies concentrate more on distributing the load evenly across the servers so that the response time is faster. But, they have not concentrated much upon the power consumption and other effects like cost based on it. Due to lack of consideration on these, providers would be required to spend more on resource provisioning thereby transferring the burden on to the customers. This results in reduction of market for the services as well as lesser ROI (Return on Investment) for the providers.

The ability of the load balancers to maintain resource utilization to a balanced level across the servers in the cloud datacenters can reduce the amount of energy consumption by avoiding overheating [3] of the nodes/servers. Energy consumption reduction would lead to reduced carbon emissions thereby supporting green computing. Increased utilization of the same resource without load balancing may result in failure of the resource (servers) thereby creating more problems to the providers from the perspective of the environment pollution due to carbon impact as well as the TCO (Total Cost of Ownership). There is also pressure from the government towards green computing initiatives,

indicating to maintain the level of datacenter emissions to a moderate permissible level in order to comply with its sustainability and licensing issues.

Thus the authors propose an enhanced load balancing algorithm by considering dynamic load and the performance factors into account in order to provision the resources in a balanced manner by improving the response time, resource utilization, energy consumption and the cost. This mechanism would prove to be most efficient in fulfilling the requirements from the providers and hence from the consumers perspective.

The remainder of the paper is divided into the following sections. Section II describes on the related work. Section III details the problem scenario. Section IV discusses the algorithm and its implementations Section V presents the result and its analysis and Section VI concludes the paper.

RELATED WORK

There are many algorithms proposed by lots of researchers to perform load balancing in cloud computing. We have many static and dynamic algorithms each having their own benefits and limitations. Among them static algorithms are less preferred because of their inability to be effective in resource utilization and fault tolerant capabilities. So, we have considered discussing about some of the dynamic load balancing strategies and their findings.

Jitendra Bhatia et. al [4]presented a dynamic load balancing strategy which considers the load and performance as main factors in order to dynamically generate a queue of servers to be provisioned for the job requests. Through experimentation they have proved that HTV algorithm results in better response time, resource utilization and fault tolerance. But, they have not concentrated upon power consumption and cost optimizations. Shreenath Acharya et. al [2] have proposed an enhanced algorithm that proved to be an enhancement for the HTV algorithm by providing some higher level of benefits in performance as well as resource utilization using user based priority. This work is an extended version of this paper.

Saurabh Jain et. al [5] have proposed an enhanced dynamic load balancing meachism which contains selection of target host and virtual machines for execution of the tasks by setting appropriate thresholds for minimzing the power cosnumption. Through simulations they had proved that their methodology results in better resource utilization and reduced energy consumption than the existing ones.

Anton Beloglazov et. al [6] have developed a threshold based approach for energy efficiency in cloud computing. They have stated that power consumption could be modelled based on the cpu utilization and an idle server consumes 70% of its maximum power consumption. They have modelled energy as function of power consumption over time. They developed a cost model using cpu utilization, amount of utilized memory and the network bandwidth.

Akash Jain et. al [7] have proposed a load balancing algorithm that uses categorization of the users based on their request. This algorithm specifically reserves resources for the users with higher priority and the normal users would be utilizing the resources as per its availability. But. They there is no information related to the utilization of the resources when the priority users are not active and it is not implemented.

Ranesh Kumar Naha et. al [8] have developed broker based algorithms to support trade-off between cost and the performance. Through simulations carried over different cloud scenarios they have shown that their approach is cost effective and results in higher performance compared with the service proximity based routing algorithm But, they have not considered resource utilization and power consumption parameters in their experimentations.

Pandaba Pradhan et. al [9] have compared the three basic load balancing algorithms like round robin, SJF and FCFS. They have informed that round robin would become FCFS if the time quantum is high and SJF is similar to FCFS with only change as SJF selects shortest job first. They proposed a modified round robin algorithm to improve the performance by dynamically changing time quantum using total cpu time and the number of jobs.

Subhashish Mohapatra et. al[10] have conducted experiments on various load balancing algorithms like round robin, FCFS, Throttled, equally spread current execution load in order to compare their efficiency. Their study & the results obtained reveal that among these four round robin is the best method with respect to the performance as well as the cost.

Jyotiska nath Khasnabish et. al [11] have introduced a mechanism of resource allocation for multi-tier cloud systems. They applied monitoring at each tiers rather than as a whole for resource allocation which proved to be more useful to improve resource utilization. But they have not considered the factors like cost, performance and power consumption.

Weiwei Lina et. al [12] have proposed a dynamic resource allocation strategy in which initially an application is assigned with a VM which satisfies its minimum resource requirements. When the load varies the interval between new allocations of the resources are set as slower change meaning longer interval and faster change means shorter interval. If the load has a range of repitions, then resource is allocated based on the changes. In order to reduce the overhead, they had set threshold value to get proper time of reallocation. This method, proved to efficient for peak load and resulted in less cost.

Sivadon Chaisiri et. al [13] have developed an efficient resource provisioning algorithm named OCRP (Opimal Cloud Resource Provisioning) specifically concentrating on minimizing the total cost . They introduced 2 plans for each user as on-demand and reservation. The provisioning phases were reservation, expanding and on-demand and each provisioning stage involved one or more provisioning phases. Through their experimentations they have shown that OCRP results in better reservation and on-demand allocation of the

resources with reduction in cost.

Saraswathi AT et. al [14] have developed a cloud provisioning strategy mainly focussing on utilization of the resources. Their mechanism used pre-emption capability thereby pausing the execution of low priority jobs when a higher priority job arrives. This method provided better utilization for higher priority jobs but, may result in increased response time for the lower priority jobs and also starvation for resources.

PROBLEM SCENARIO

In the cloud computing environment, whenever the user's job is submitted it must be assigned with the appropriate resources for execution. The allocation of the best resources for the effective and efficient execution is the main reason behind the success of this technology. Apart from this, maintaining the healthiness of the resources from the datacenter is also a major concern from the providers side in order to have significant benefits from the economic perspective. Thus, load balancing becomes a prime concern to be considered for cloud provisioning. To achieve this, the following scenario had been considered.

Let C be the cloud environment represented as C = {D_1, D_2, D_3,…,D_n) where D_1,D_2,D_3,…,D_n are the number of datacenters in the cloud by the providers.

Each datacenter has m number of servers given by,

D = {S_1,S_2,S_3,….,S_m} where m>n

Each server may contain l number of virtual machines,

i.e., S={v_1,v_2,….,v_l} where l>m

Let J= {j_1,j_2,j_3,….,j_k} be the jobs submitted for the execution in cloud environment. where k>=l

The problem is to optimally balance and schedule the jobs J in C such that resources in S are utilized in a balanced manner resulting in lesser response time and also reduced power consumption.

i.e., $C= \sum_{i=1}^{m} Di$ (1)

$D =\sum_{j=1}^{n} Sj$ (2)

Using 1 & 2, V ϵ S must be able to execute the task efficiently ∀J.

For ex: Consider figure 1 containing 2 servers. If the numbers of jobs are 5 and the numbers of virtual machines in each server are 3. All the VMs are configured with equal Mips and RAM capability. When the jobs are submitted, they would be assigned to the first fit VMs from the servers if they are not scheduled in a balanced manner. Due to this most of the assigned jobs would be executed in the resources of the server1 making it overloaded. This is because, once the previously assigned jobs are finished by a VM, again the incoming jobs are assigned to it based on its fitness and the same process continues. In the real time scenario, it would make server1 more prone to failure owing to its excessive usage and may result in its breakdown. Server2 would be utilized very less although it is capable of executing the jobs. This results in more time and hence lesser performance by the cloud. According to the studies, it has been mentioned that an idle server consumes 70% of maximum power consumed by a server. Instead in these situations it would be working and utilized to a least extent but consumes significant amount of energy.

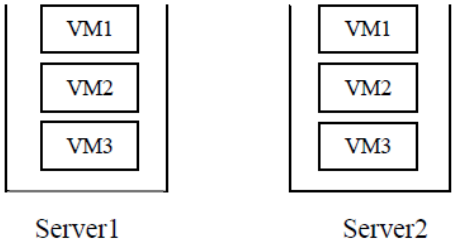

Figure 1: Problem Scenario

SYSTEM ARCHITECTURE

The basic architecture of the proposed system is shown in fig.2. It consists of the 4 main components as clients, jobs with user priority, dynamic load balancer and the resource pool.

Client: This entity requests the cloud services through job inputs. Each job could be assigned with a jobtype or priority indicating its requirement of faster, moderate or lower speed of executions. It facilitates the load balancer with priorities for each submitted job in order to comply with users' requests. The overall execution of tasks mainly depend upon the clients as per their wish to have faster or normal completion of the tasks.

Dynamic Load Balancer: It is the main component which performs the load balancing using 2 main components namely, scheduler and the resource allocator. Both these components work hand in hand to accomplish the task.

i). Scheduler: It accepts the incoming job requests as per their priority and sorts the requests according to it. It performs sorting at both the VM level and the cloudlet level. At the VM level, it sorts the VMs based on their mips value. At the cloudlet level, the jobs are sorted according to their cloudlet length at the individual priority level group requested by the clients. Once all the sorting are completed a scheduling list is prepared and it is given to the resource allocator.

ii). Resource Allocator: It receives the scheduled request from the scheduler. It assigns the resources i.e., hosts and their corresponding VMs according to the schedulers request such that there won't be any possibility of overloading/over

burdening of the hosts while fulfilling the requirements. It maintains a queue of temporary and permanent resources dynamically updated as per the jobs arrival. The resources are taken from the queue after identification of their capability to fulfill the new arrivals. Resource allocator avails the resources from the resource pool of the cloud providers.

Resource Pool: It is the datacenter hosting n number of resources which could be utilized in a shared manner using the virtualization techniques. The resources could be servers, networking infrastructures, storages since the context considered here is from the IaaS providers perspective. Apart from this, figure 2 shows only servers in the resource pool because we have considered server i.e., computational machine as a main component for our work.

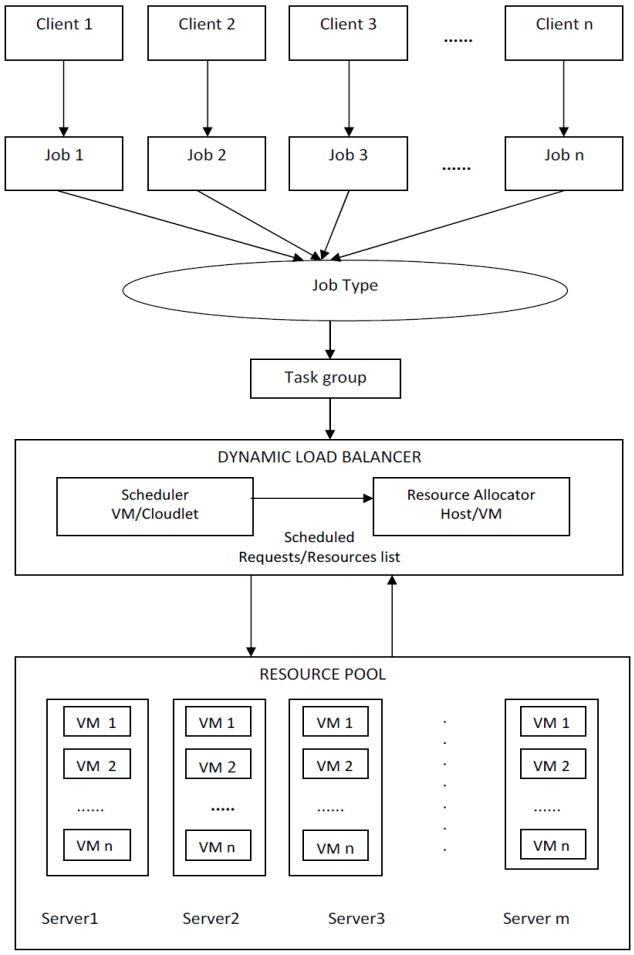

Figure 2: Basic System Architecture

The Resource Utilization of each server is given by:

$U_S = U_p + U_r/2$ (3)

U_p is the utilization of servers CPU capacity

U_r is the utilization of servers RAM capacity

$U_{avg} = \sum_{i=1}^{N} Usi/N$ (4)

N is the no. of servers

Power consumption of the server based on [6] is given by,

$U_p = U_p * \alpha$ if $U_P < U_{opt}$ and $U_p > 0$ (5)

$U_p = U_p * \alpha + (U_{opt} - U_P)^2 * \beta$ if $U_P > = U_{opt}$ and <1 (6)

P_{min} is the power consumed when the resource (server) is idle and P_{max} is the maximum power consumed when the server is active with load. The normalized utilization is a factor calculated based on the identification of optimum utilization, U_{opt} as 0.7 and assigning its value based on the non-linear relationships between the utilization and the dynamic power by setting the values of α and β are set as 0.5 and 10 respectively [15].

$P_s = P_{min} + (P_{max}-P_{min})*U_p$ (7)

$P_{overall} = \sum_{i=1}^{N} Psi$ (8)

Dynamic energy consumption is modeled as power consumption over a period of time T given by,

$E = \sum_{t=0}^{T} Poverall$ (9)

Cost of utilizing resources is calculated by considering the amount of resources utilized like MIPS and RAM for execution of all the tasks. Thus it is mainly calculated based on the cost of energy consumption by excluding the constant factors like cooling and others.

Cost, C for resource provisioning is given by,

$C = \text{Cost per KWh}(E + E_{cooling})$ (10)

Cost per KWh is considered as per the rates of the leading cloud provider Amazon [17] and $E_{cooling}$ is a constant for racks.

IMPLEMENTATION

The implementation of the proposed strategy is carried out in cloudsim using java programming language and eclipse editor. The user interface has been designed to dynamically input the datacenter configuration by way of number of servers & their configurations. Number of VMs and their configurations could also be configured as per the need.

Apart from this, it permits the users to describe their job in the form of cloudlet lengths and also provisions for assigning priorities for each job. These informations will be useful for the clients to mention their requirements during the task/job submissions. It would also enable the scheduler to set the scheduling and load balancing according to the assigned priorities.

The following pseudocode describes the steps in initial setting and reconfiguration of the resources as per the proposed scheme.

Initialize the datacenter with S servers

$S = \{s_1, s_2, s_3, \ldots, s_n\}$

Create m number of VMs

sort (mips, vmlist)

```
Input: No of tasks, type p
Task groups: t1, t2, t3
For each request  from user
no_of_tasks_per_vm= no_of_tasks /m
case: p=1
{
  assign t to t1
  sort (length,tasklist)
}
case:  p=2
{
  assign t to t2
  sort (length,tasklist)
}
default:
{
 assign t to t3
 sort (length,tasklist)
}
EndFor
Foreach vm do
while ( ! no_of_task_per_vm)
task++
map task to vm
no_of_tasks_per_vm = no_of_tasks_per_vm -1
EndWhile
EndFor

while (nextUnassignedTask)
Foreach vm do
task= nextUnassignedTask
map task to vm
EndFor
EndWhile
```

The proposed mechanism generates a dynamic queue of load [2,4] across the servers in order to have an understanding of which servers could be assigned with the next task while maintaining the load balancing. This uses specifically the creation of a temporary queue every time based on the performance and load factors calculated. The algorithm shown below describes the process of generating dynamic queue for improved load balancing.

```
lf = Total  no. of resources – no. of used resources
pf1 = average (current _response _time)
pf = pf1 - (previously calculated pf1)
pf =pf/(previous pf1)*100
load =lf - pf
If (load < 0)
Assign load = 0
EndIf
Min _factor= min (all loads)
Z = load / Min _factor
Generate Dynamic Queue on base of Z
Assign jobs based on Z for Servers
```

Table1 describes an example of the initial configuration of the 10 virtual machines along with their mips rating for the sample configuration. Each of the 20 cloudlets are submitted with job length (cloudlet length) and the corresponding type specified by the clients based on their requirements of the performance.

Table 1: Initial Configuration

VM ID	MIPS	Cloudlets			Cloudlets		
		ID	Length	Type	ID	Length	Type
1	500	1	10000	1	11	3000	3
2	300	2	20000	1	12	9000	2
3	200	3	15000	1	13	5000	3
4	400	4	5000	3	14	2000	3
5	250	5	8000	2	15	16000	1
6	350	6	4000	3	16	6000	2
7	450	7	7000	2	17	8000	2
8	150	8	6000	2	18	4000	3
9	600	9	11000	1	19	10000	1
10	550	10	13000	1	20	3000	3

Table 2 describes the re-arranged configurations before the execution of the tasks. It performs sorting the virtual machines according to their MIPS value. It also shows the grouping of the tasks based on their type as requested by the client. Inside the groups, again the cloudlets are sorted according to their lengths.

Table 2: Task grouping

VM ID	MIPS	Cloudlets			Cloudlets		
		ID	Length	Group	ID	Length	Group
1	600	1	20000	1	11	7000	2
2	550	2	16000		12	6000	
3	500	3	15000		13	6000	
4	450	4	13000		14	5000	3
5	400	5	11000		15	5000	
6	350	6	10000		16	4000	
7	300	7	10000		17	4000	
8	250	8	9000		18	3000	
9	200	9	8000		19	3000	
10	150	10	8000		20	2000	

When multiple tasks have same length they are sorted based on FCFS as per their ID value. Virtual machine sorting also follows the same procedure. The mappings of the VMs to the cloudlets are carried out by calculating the value of number of cloudlets per VM. In this case, it would be 2 cloudlets per VM. Load balancing algorithm is applied based on these mapping possibilities by generating a dynamic queue to identify the load on a server to decide whether to add a job/task to it or not i.e., to identify how many jobs could be added to this server.

RESULTS AND DISCUSSION

The experimentation has been conducted in cloudsim using java programming language with Eclipse editor. The user interface has been designed such that it allows the client to input their job requests with specific types. It also permits configuring number of servers and number of virtual machines such as mips, ram, number of cores and processing elements appropriately.

The datacenter(s) are configured with both homogeneous and heterogeneous servers for our experimentation purposes and the parameters considered are response time, resource utilization, energy consumption and the cost along with load balancing capability.

The VMs are configured with mips values as 200, 300, 400 and 500 repeated in a random order as per the considered number of VMs. The ram values are set as 256MB and 512MB respectively in each case. The cloudlets are configured with their lengths with values from 1000 to 30000 utilized randomly.

A. Homogeneous Servers

In the homogeneous configuration, the servers are set with 1000MIPS and 1024MB RAM along with number of processing elements and the number of cores set to one. The numbers of VMs are set as 50, 100, 150, 200, 300 respectively. The workload is configured with varieties of tasks pertaining to various application types as 100, 200, 300, 400 and 500.

Figure 3 shows the performance of the proposed algorithm with respect to the existing load balancing policy named HTV. It is evident from the figure that the proposed scheme results in better performance in terms of the response time and it has been found that it is able to the achieve about 13.6% improvement in performance.

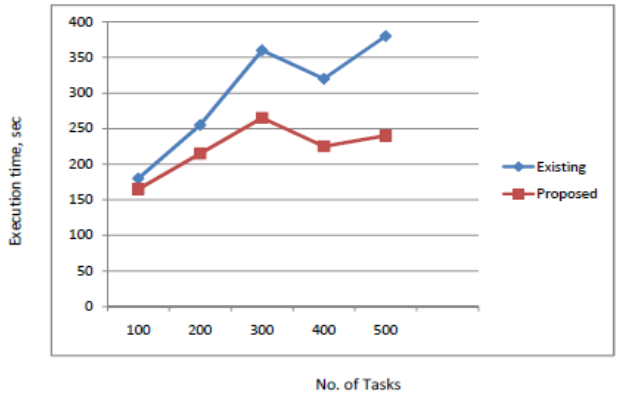

Figure 3: Execution time Vs. No. of Cloudlets

Further experimentation has been conducted for various datacenter configurations having varied number of servers. Figure 4 depicts the average percentage resource utilization for several datacenter configurations. It can be inferred from the figure4 that the resource utilization of the proposed scheme is better than the considered existing policy.

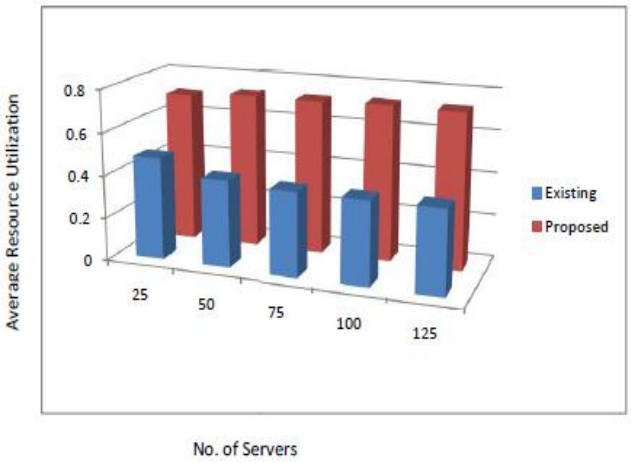

Figure 4: Resource utilization Vs. No. of Servers

It is found that the proposed system is able to achieve about 22.8% increase in resource utilization compared to the existing system.

The energy consumption of the proposed system is compared with the existing algorithm in figure5. The values of the energy consumption under different datacenter configurations with varied jobs show that the proposed strategy consumes about 10.22% less energy than the existing system.

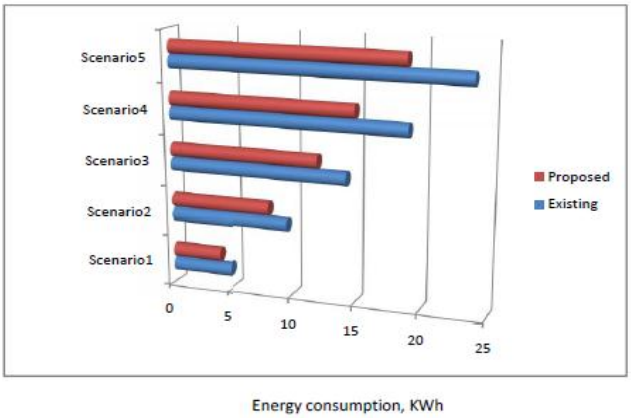

Figure 5: Energy consumption at various datacenter configurations

The cost incurred for executing the assigned tasks based on their priority is presented in figure 6. It shows that the proposed scheme results in about 10.26% lesser burden in terms of cost, thereby leading to reduced expenses for both the providers and the consumers.

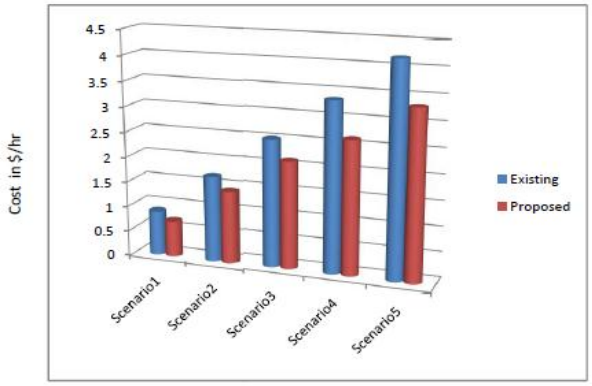

Figure 6: Cost Vs. Data center configurations

The load balancing capability for homogeneous servers is measured based on their utilizations. The resource utilization (u) is compared based on the values as very low, low, moderate and high. Its mapping is as follows:

Very low	if u <=0.2
Low	if u >0.2 && u<=0.3
Moderate	if u >0.3 && u<=0.6
High	if u>0.6

The load balancing is categorized as good, moderate and low respectively. Good corresponds to the amount of load across the servers with minimal variations (less than 1%) from the average load, moderate means loads are not same across the servers but within the permissible tolerance of 1 to 10% from the average load. Low means uneven load across all the servers. Table 3 shows the comparison of load balancing and utilization of proposed scheme and the existing system.

Table 3: Comparison of Resource Utilization and Load Balancing for datacenters with homogeneous server configurations

Datacenter Configurations	Resource Utilization		Load Balancing	
	Existing	Proposed	Existing	Proposed
Scenario1	Moderate	High	Moderate	Moderate
Scenario2	Moderate	High	Good	Good
Scenario3	Moderate	High	Moderate	Good
Scenario4	Moderate	High	Moderate	Good
Scenario5	Moderate	High	Moderate	Good

It could be noted from table 3 that the proposed system is able to achieve higher utlization. Load balancing capability is moderate under different scenrios for existing algorithm but, it is good in the proposed algorithm thereby showing its consistency across various datacenter configurations and workloads.

B. Heterogeneous Servers

In these configurations, the servers are set with 1000MIPS and 2000MIPS. The ram sizes are set as 1024MB and 2048MB respectively.

The numbers of VMs and their configurations as well as the workloads are set similar to the settings of the homogeneous servers.

Figure 7 depicts the response time of the proposed system to be considerably better than the existing HTV load balancing policy. It is able to achieve about 16.97% reduction in response time thereby shows improved performance.

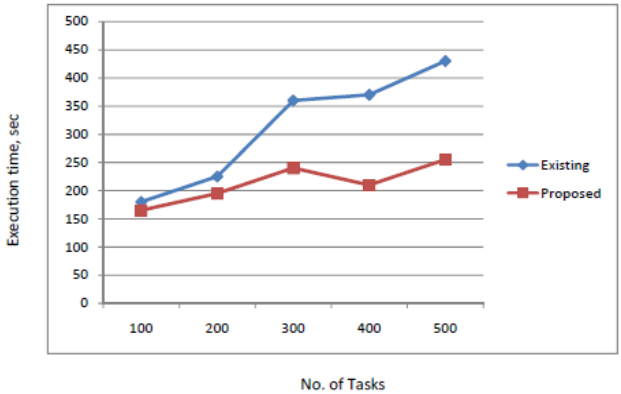

Figure 7: Execution time Vs. No. of Cloudlets

The resource utilization for various datacenter configurations is shown in figure 8 which clearly shows the remarkable differences of about 27.78% higher utilization compared with the existing policy thereby revealing its effectiveness.

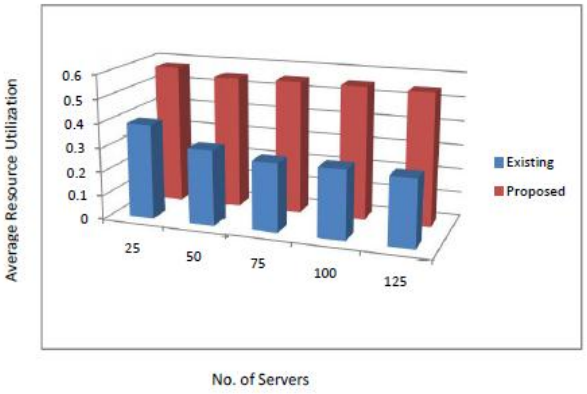

Figure 8: Resource Utilization Vs. No. of Servers

The energy consumption in case of heterogeneous servers in the datacenter has been displayed in figure 9. It clearly presents the supremacy of the proposed algorithm with respect to the existing policy. It is able to achieve about 6.45% reduction in energy consumption.

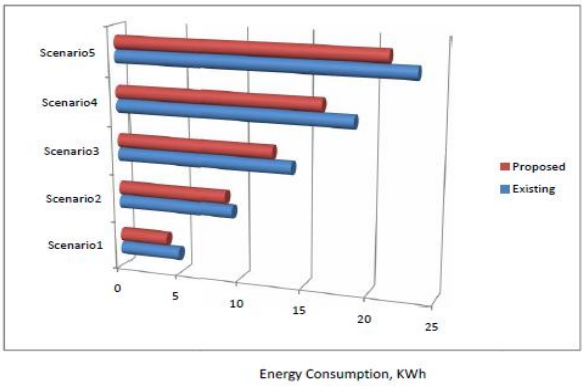

Figure 9: Energy Consumption Vs. Datacenter configurations

The cost of resource consumption is compared and shown in figure 10. This graph shows that the cost incurred for the execution of the tasks is about 6.4% lesser in the proposed policy.

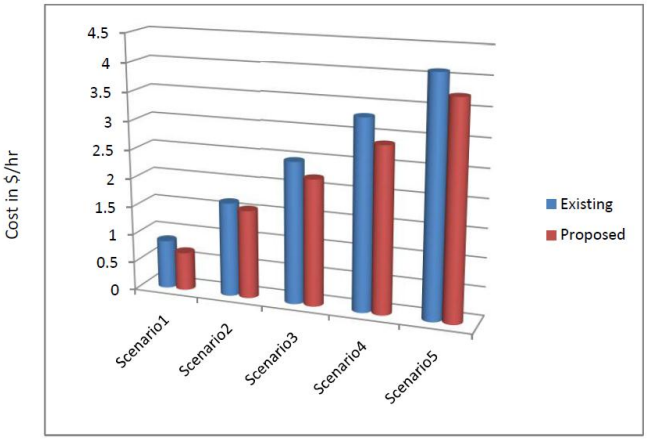

Figure 10: Cost Vs. Datacenter Configurations

Table 4 shows the comparison of effectiveness of the proposed policy with respect to the existing policy in terms of resource utilization and the load balancing capabilities. It follows the same mapping as considered for the datacenters with homogeneous servers described earlier.

It can be seen from table 4 that the proposed scheme clearly out performs the exitsing mechanism in both the resource utilization and the load balancing ability.

Table 4: Comparison of Resource Utilization and Load Balancing for datacenters with heterogeneous server configurations

Datacenter Configurations	Resource Utilization		Load Balancing	
	Existing	Proposed	Existing	Proposed
Scenario1	Moderate	Moderate	Moderate	Moderate
Scenario2	Moderate	Moderate	Good	Good
Scenario3	Low	Moderate	Moderate	Good
Scenario4	Low	Moderate	Moderate	Good
Scenario5	Low	Moderate	Moderate	Good

CONCLUSION AND FUTURE SCOPE

The main issues of consideration in the area of cloud computing like response time, resource utilization, energy consumption, cost and the load balancing are considered to measure the capability of the proposed scheme. After testing the obtained results under homogeneous and the heterogeneous servers based datacenter configurations it can be clearly inferred that the proposed algorithm is better than the considered existing algorithm since it results in lesser response time, improved resource utilization, lower energy consumption and low cost. The load balancing ability is also found to be very promising compared with the existing algorithms thereby proving it to be more efficient and effective. It could also be noted that increase in number of tasks does not incur in significant increase in the response time as per the variations in the workloads at both the homogeneous and heterogeneous datacenter configurations.

The future scope could be testing the proposed algorithm in real time cloud set up to further study and analyze its effectiveness..

REFERENCES

[1] Amanpreet Kaur, Bikrampal Kaur and Dheerendra Singh,"Optimization Techniques for Resource Provisioning and Load Balancing in Cloud Environment: A Review", *I.J. Information Engineering and Electronic Business*, MECS Publishers, 2017, 1, 28-35, DOI:

10.5815/ijieeb.2017.01.04

[2] Shreenath Acharya and Demian A. D'Mello, "Enhanced Load balancing algorithm for Resource Provisioning in Cloud", Proceedings of the IEEE International conference on Inventive Computation Technologies (ICICT-2016), Coimbatore, ISBN: 978-1-5090-1285-5, DOI: 10.1109/INVENTIVE.2016.7824851

[3] Foram F Kherani and Jignesh Vania ,"Load Balancing in cloud computing", - International Journal of Engineering Development and Research (IJEDR), Volume 2, Issue 1, pp. 907-912, ISSN: 2321-9939, 2014

[4] Jitendra Bhatia, Tirth Patel, Harshal Trivedi and Vishrut Majmudar, *"HTV Dynamic Load Balancing Algorithm for Virtual Machine Instances in Cloud"* 2012 International Symposium on Cloud and Services Computing, ISBN: 978-0-7695-4931-6/12, DOI 10.1109/ISCOS.2012.25, IEEE

[5] Saurabh Jain and Varsha Sharma, "Enhanced Load Balancing Approach to Optimize the Performance of the Cloud Service using Virtual Machine Migration", *I.J. Engineering and Manufacturing*, MECS Publishers,1, 41-48, 2017, DOI: 10.5815/ijem.2017.01.04

[6] Anton Beloglazov and Rajkumar Buyya,"Adaptive Threshold Based Approach for Energy Efficient Consolidation of Virtual Machines in Cloud Data Centers", MGC 2010, ISBN: 978-1-4503-0453-5/10/11, ACM

[7] Akash Jain and Pinal Patel, "Load Balancing in Cloud Computing", International Journal of Engineering Development and Research (IJEDR), Vo.3, Issue 2, pp: 929-936, ISSN:2321-9939, 2015

[8] Ranesh Kumar Naha and Mohamed Othman, "Cost-aware service brokering and performance sentient load balancing algorithms in the cloud", Journal of Network and Computer Applications, 75(2016), 47–57, Elsevier, 2016 DOI: 10.1016/j.jnca.2016.08.018

[9] Pandaba Pradhan, Prafulla Ku. Behera and B N B Ray, "Modified Round Robin Algorithm for Resource Allocation in Cloud Computing", International Conference on Computational Modeling and Security (CMS 2016), Elsevier Procedia Computer Science 85 (2016), 878 – 890, 2016

[10] Subasish Mohapatra, K.Smruti Rekha and Subhadarshini Mohanty "A Comparison of Four Popular Heuristics for Load Balancing of Virtual Machines in Cloud Computing", International Journal of Computer Applications (0975 – 8887,) Volume 68– No.6, 2013

[11] Jyotiska Nath Khasnabish, Mohammad Firoj Mithani and Shrisha Rao, "Tier-Centric resource allocation in multi-tier cloud systems", IEEE Transactions on Cloud Computing, in press, Citation information: DOI 10.1109/TCC.2015.2424888

[12] Weiwei Lina, James Z. Wangb, Chen Liangc and Deyu Qi, "A threshold-based dynamic resource allocation scheme for cloud computing", Published by Elsevier Ltd, Procedia Engineering 23 (2011), 695 – 703, 2011

[13] Sivadon Chaisiri, Bu-Sung Lee and DusitNiyato, "Optimization of resource provisioning cost in cloud computing", IEEE transactions on services computing, Published by the IEEE Computer Society, Vol. 5, No. 2, 2012

[14] Saraswathi AT, Kalaashri.Y.RA and S. Padmavathi, "Dynamic resource allocation scheme in cloud computing", Published by Elsevier B. V, Procedia Computer Science, 30 - 36, 2015

[15] Yue Gao, Yanzhi Wang, Sandeep K. Gupta, and Massoud Pedram, "An Energy and Deadline Aware Resource Provisioning, Scheduling and Optimization Framework for Cloud Systems," Proceedings of the Ninth IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis, Piscataway, NJ, USA, 2013, pp. 31:1–31:10.

[16] Shreenath Acharya and Demian Antony D'Mello," A taxonomy of Live Virtual Machine (VM) Migration mechanisms in cloud computing environment", Proceedings of IEEE International Conference on Green Computing, Communication and Conservation Energy, ICGCE 2013, pp. 811-817, ISBN: 978-1-4673-6126-2, DOI: 10.1109/ICGCE.2013.6823545

[17] https://www.datacenters.com/news/infrastructure/135-data-center-power-costs-and-requirements. visited on 12 Dec 2017