Achieving Performance Optimization by Enhancing Energy Efficiency in Green Cloud Computing

A Ph.D. Synopsis submitted to

Gujarat Technological University, Ahmedabad

for the Award of

Doctor of Philosophy

in

Computer/IT Engineering

by

Hiteshkumar Arajanbhai Bheda

Enrollment No. 159997107011

under the supervision of

Dr. Chirag Suryakant Thaker

Professor and Head,
Computer Engineering Department,
Government Engineering College, Rajkot, Gujarat, India

DPC Member 1
DR. SANJAY M. SHAH
Professor and Head
Computer Engineering Department
L.D. College of Engineering, Ahmedabad

DPC Member 2
DR. DARSHAN B. CHOKSI
Professor
G.H. Patel Institute
S.P. University, Vallabh Vidyanagar

GUJARAT TECHNOLOGICAL UNIVERSITY, AHMEDABAD

October - 2021

INDEX

1. Title of the thesis and Abstract

Title of the thesis: Achieving Performance Optimization by Enhancing Energy Efficiency in Green Cloud Computing

Abstract

A global digitization advancement and deployment of a huge number of mobile and web applications increase the need for cloud-based solutions and data centers. A rise in the number of internet users also requires more data centers. Increasing demand of data centers simultaneously needs a huge amount of energy for data center operation and on another end emits an enormous amount of CO_2. Several approaches have been proposed by researchers to reduce energy consumption. The problem with existing work is it compromised with important factors like Quality of Service and system performance. The research hypothesis proposes three methods: Task submission and VM placement, Load-Aware VM placement, and Overloaded / Underloaded host selection. Task submission and VM placement method improves overall response time as well as data center processing time that helps to optimize system performance. Load-Aware VM placement method significantly decreases VM migration and increases mean time before VM migration that in turn helps to reduce energy and associated cost. Adapting queue-based mechanism in Load-Aware VM placement method achieves complete workload execution to gain QoS and to retain SLA in cloud system. Overloaded / Underloaded host selection method identifies overloaded and underloaded hosts. This method neither allows assigning additional workload nor grants permission to place VM if the host is either overloaded or underloaded, which improves QoS and reduces energy consumption for hosts. Two stated methods, first task submission and VM placement, and second Load-Aware VM placement aim to reduces VM migration, saves energy, and increases the value of mean time for better performance. Overloaded / Underloaded host selection method stated to improve system performance and maintain SLA.

2. Brief description on the state of the art of the research topic

Cloud computing is famous for its potential capabilities and offers computing resources like CPU (Processing Power), Memory, Storage, Network, and many more as a utility. According to NIST [1] definition, cloud computing can be considered as an on-demand and convenient model for computing resources with fast provisioning and offered with little interference of management or service provider interaction. Prominent cloud service providers; like Amazon through Amazon Web Services (AWS), Google thru Google Cloud Platform (GCP), Microsoft via Microsoft Azure, IBM, Alibaba cloud; are performing well and investing great efforts to extend the capabilities in different fields. Conceptually Green Cloud Computing handles cloud resources that can deal with energy parameters. It is a vital solution for all the crucial dynamic resources demand with scalability.

The number of internet users is growing rapidly hence there is a necessity to upscale data centers in different zones and geographical regions across the globe. According to the report submitted to the U.S. Department of Energy in 2014, 1.8% of total energy is consumed by data centers in the USA [2]. Data centers emit an enormous amount of CO_2 [3] which is an extremely challenging problem for the world which may lead towards global warming threat. Due to this major issue, reduction in energy consumption should be the key concern. Resolution to this issue will help sustain a green environment and assist leading organizations to grow dramatically.

Various areas are researched and proposed to work on energy-proficient methodology in cloud environment due to trust factors of cloud computing. Anton Beloglazov et al. [4] has suggested the taxonomy of the energy efficient architecture of computer systems such as hardware, operating system, virtualization, and data centers. David Aikema et al. [5] presented a VM migration strategy with two distinct approaches; in the first approach, authors calculated migration time and related energy consumption for the migration, while in the second approach they were directly migrating all the workload to available other VMs based on threshold values. The authors have made a correlation with different scenarios where it has been inferred that ascertaining migration time and required energy utilization computation is much advantageous.

In one of the mechanisms for VM migration [6], Minu Bala and Devanand have focused on the emission of CO_2 and VM migration based on CPU utilization factors. They adapted upper-threshold and lower-threshold of CPU limit, based on which VMs are migrated to different data centers on respective hosts. K. Zhang et al. [7] have reflected on the matter of virtual machine (VM) energy saving options on an overloaded host in a cloud computing environment. During a VM migration, they studied the energy influencing factors and built energy efficient VM selection techniques rely on greedy algorithms and the process of dynamic programming. Junaid Shuja et al. [8] provided a study of green computing methods in the context of contemporary IT tools in our community. This addresses effective strategies for green computing and the trade-off between green and high-performance initiatives. They also analyzed the inevitable issues that expanding IT technologies face in their effective green operational activities.

Ali Hammadi et. al. [9] has classified the current Data Center Network (DCN) architectures into switch-centered and server-centered topologies, as well as established various techniques to reduce energy usage. Yacine Kessaci et al. [10] have proposed an Energy-aware Multi-start Local Search (EMLS-ONC) algorithm to optimize energy consumption for OpenNebula-based Cloud. Y. Luo et. al. [11] have proposed a general energy consumption optimization algorithm for Service Level Agreement (SLA) cloud workflow programming, which reduces energy usage and hence fulfills the time and expense efficiency criteria.

Huankai Chen et al. have provided a User-Priority Guided Min-Min Scheduling algorithm [12]. The authors have provided two distinct techniques Load Balance Improved Min-Min (LBIMM) and Priority Awared Load Balance Improved Min-Min (PA-LBIMM). If we compare the results of LBIMM and PA-LBIMM with Min-Min algorithm then they are quite better to decrease tasks' completion time, proper load balancing of resources, and enhance overall system performance. This improved algorithm increases system performance by over 20% for the utilization of resources and user services. Veerawali Behal et al. have proposed Load Balancing technique for Heterogeneous Environments in cloud computing [13]. Authors have proposed throttled technique that can deal with optimized response time. This technique offers performance enhancement in distributed cloud environment using broker policy.

Kunjan Garala et al. have provided Performance Analysis using Dynamic algorithm [14]. Dynamic algorithm usually works on distinct parameters, like Response Time, Resources Utilization, Fault Tolerant, Waiting Time, Throughput, Turnaround Time, Process

Migration, and Stability, and because of its dynamicity characteristics, it provides good performance compared to a static algorithm. B. Kruekaew et al. have proposed a method of Artificial Bee Colony [15]. ABC_LJF method provides efficient results compared to other methods and provides high performance in terms of system scalability. In ABC algorithm, scheduling is executed with consideration of tasks' size and Longest Job First (LJF) to reduce the makespan of data processing time.

Lin Wang et al. [16] have established a time-conscious model for energy efficiency and resolved the problem of assigning virtual machines to servers for reduced traffic and favorable conditions for traffic design. This strategy further decreases the number of active switches and synchronized flows to estimate the energy consumption of a virtual machine based on in-processor events. To prepare a VM according to the energy allocation, the Energy-Credit Scheduler (ECS) has been introduced instead of CPU time which relies on energy consumption for a specific VM.

Akindele A. Bankole and Samuel Adesoye Ajila [17] have evaluated mathematical models such as Linear regression along with the most used classifier Support Vector Machine in amalgamation with Artificial Neural Networks (ANN). The researchers have projected that these methods helped to find out the optimal solution in terms of response time and consumption variables.

Yunliang Chen et al. [18] proposed a two-tier VM placement framework with a feasibility-driven stochastic VM placement (FDSP) algorithm which to reduce energy consumption for VM placement process. The proposed algorithm saves 15.3% energy consumption compared to general algorithms while reduces 15.7% cost of energy compared to other VM placements policies.

Sourav Kanti Addya and Anurag Satpathy [19] proposed a Game theoretic approach to figure out the best cost for VM placement in cloud-based data centers. Authors have considered n-users with a cooperative game environment to identify precise value for capital expenditure. For energy-efficient placement policy, integer linear programming is proposed and Microsoft Azure and Amazon EC2 have been considered as cloud service providers. In concluding remarks, with regards to consuming lower energy: the authors' proposed game thematic approach is more effective compared to first fit decreasing (FFD) and enhanced first fit decreasing (EFFD).

3. Definition of the Problem

- Most of the research has been executed on scheduling, job placement, or using particle swarm optimization (PSO) algorithms to reduce energy consumption in Green Cloud Computing.
- There are various parameters like CPU, Memory, Bandwidth, Storage, GPU, etc., which consume energy to execute the assigned workload. However, the majority of the researchers have used either one or a combination of the parameters, and the remaining parameters might be overlooked to calculate the system's actual capacity.
- To reduce energy consumption and to optimize system performance, the allocation of VMs can be divided into two challenges:
    1. Placement of a new VM on the host
    2. Optimization of current VM's allocation
- Find out a host which has no task for execution, so turn it off or put it into power-saving mode to reduce the overall energy consumption.
- Find out the host which is overloaded or underloaded. Select VMs and choose migration strategy in case they are running over either overloaded or underloaded host.

4. Objective and Scope of the Work

Research title and proposed approaches have been carried out to achieve the following objectives:

- Minimize the energy consumption by system or resources in use.
- Improve cloud system performance.
- Maintain and improve Quality of Service (QoS).

To achieve the above-mentioned objectives, mainly three parameters; Memory, Mips, and Bandwidth have been considered.

Scope of the work includes:

- The calculation of load and utilization of resources.
- Proposed system architecture and algorithms to reduce energy consumption and optimize system performance.

5. Original contribution by the thesis

System architecture with a Load-Aware VM placement mechanism has been proposed to manage cloud resources. This system mechanism helps to optimize resource allocation, improves system performance, and reduces energy consumption.

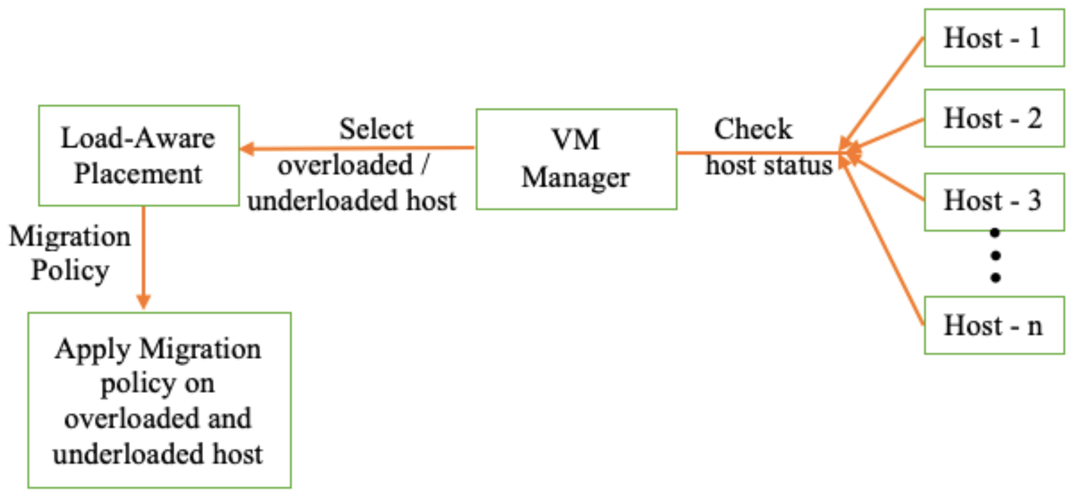


Figure 1: Proposed System Architecture with Load-Aware VM Placement Mechanism

Figure 1 illustrates the proposed system architecture with Load-Aware VM placement. There are multiple hosts available on data centers with different capacities. With the proposed mechanism, each host is being checked and updating the status to VM Manager. On the next step overloaded or underloaded host status is being assigned based on threshold values. Load-Aware VM placement algorithm starts deployment of VMs to execute workload, and it also decides to migrate VMs in case of overloaded or underloaded hosts.

Three methods, Task submission and VM placement, Load-Aware VM placement, and Overloaded / Underloaded host selection have been proposed for the research hypothesis. The proposed methods help to optimize system performance and to reduce energy consumption. Three main parameters Memory, Mips, and Bandwidth have been used for the proposed methods. Load, Utilization of host, and Utilization of VM are the key factors and they have been calculated as follows:

$$Load = \frac{Memory + Mips + Bandwidth}{3} \qquad (1)$$

$$Utilization\ of\ Host = \frac{Total\ load\ used\ by\ all\ VMs\ on\ host}{Total\ load\ of\ host} \tag{2}$$

$$Utilization\ of\ VM = \frac{Total\ load\ used\ by\ all\ tasks\ on\ VM}{Total\ load\ of\ VM} \tag{3}$$

Power Consumption is calculated with Memory, Mips and Bandwidth as follow:

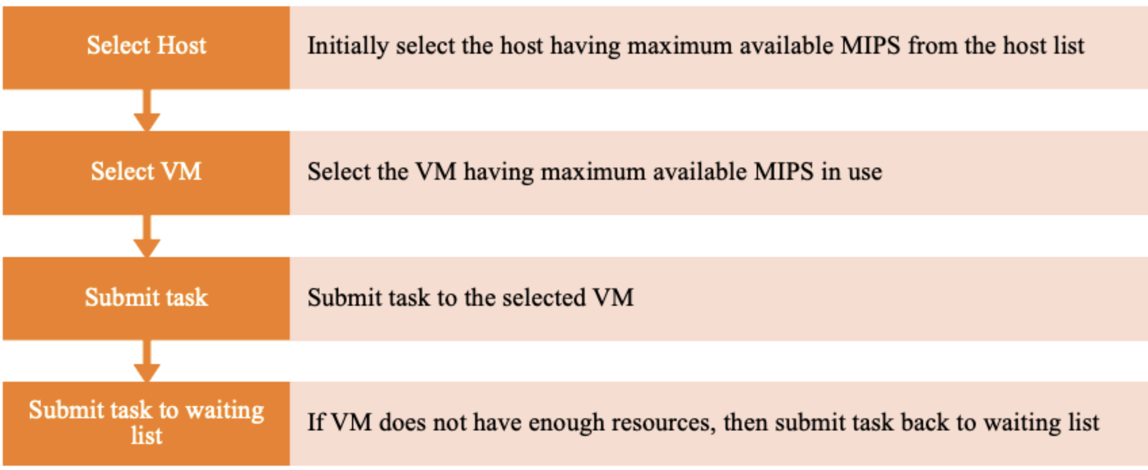
$$P_c = P_{min} + (P_{max} - P_{min}) + Utilization \tag{4}$$

$$P_c = P_{min} + (P_{max} - P_{min}) + \left[\frac{\frac{\sum_{i=0}^{n} ram\ v_i}{ram_{pm}} + \frac{\sum_{i=0}^{n} Mips\ v_i}{Mips_{pm}} + \frac{\sum_{i=0}^{n} bw\ v_i}{bw_{pm}}}{3} \right] \tag{5}$$

6. Methodology of Research, Results with Comparisons

A. Task submission and VM Placement

Task submission and VM placement method uses parameters like task length, Mips, no. of PE (Processing Elements/cores), no. of tasks, and no. of VMs. The process of task submission and VM placement can be furnished with the following steps:



Algorithm 1: Task submission and VM Placement

Input: Vm_list, Task_list, Host_list
Output: $VM_{id(j)}$ to submit task t_i

```
Host_id = null
VM_id = null

//Find Host from host k having maximum available MIPS
For each t_i task from Task_list
        Host_available_mips = 0
        for each Host k from Host_list
                if (HOST_available_mips < Host mips_k)
                        HOST_available_mips = Host mips_k
                        Host_id = host_k.host_id
                End if
        End for
End for

//Find VM from host k having maximum available MIPS
For each vm from host having id = Host_id
        VM_available_mips = 0
        for each vm j from Vm_list
                if (VM_available_mips < vm mips_j)
                        VM_available_mips = vm mips_j
                        VM_id = VM_k.VM_id
                End if
        End for
End for

If VM_id is not null and resources are available in VM_id then
        submit task to VM having id = VM_id
else
        submit task to Task_list
```

Experimental Setup for task submission and VM placement method:

For the implementation of task submission and VM placement method, we have installed and configured OpenStack over CentOS. To begin with the initial experiment, we have considered 10 hosts. 50 virtual machines were deployed with the capacity of 5 processing elements. For the input workload, we have fed 500 tasks with 5000 task length.

Overall response time and data center processing time values have been recorded using above mentioned experimental setup. Overall response time and data center processing time have been calculated for three methods, Service Proximity Based Routing, Performance Optimized Routing, and Dynamically Reconfigurable Routing. For result analysis proposed load balancing approach is implemented along with traditional approaches like Round

Robin, Equally Spreaded Current Execution, Throttled, and Least Frequency Used. The results for overall response time and data center processing time are described in Table 1 and Table 2 respectively.

Table 1: Comparative Analysis of Overall Response Time

	Proximity based Routing (ms)	Optimized Response Time (ms)	Reconfigure Dynamically (ms)
Round Robin	158.42	152.65	158.49
Equally Spreaded Current Execution (ESCE)	152.13	153.43	157.24
Throttled	152.04	152.67	156.92
Least Frequently Used (LFU)	151.56	151.66	155.8
Proposed Load Balancing Policy	150.34	151.08	155.6

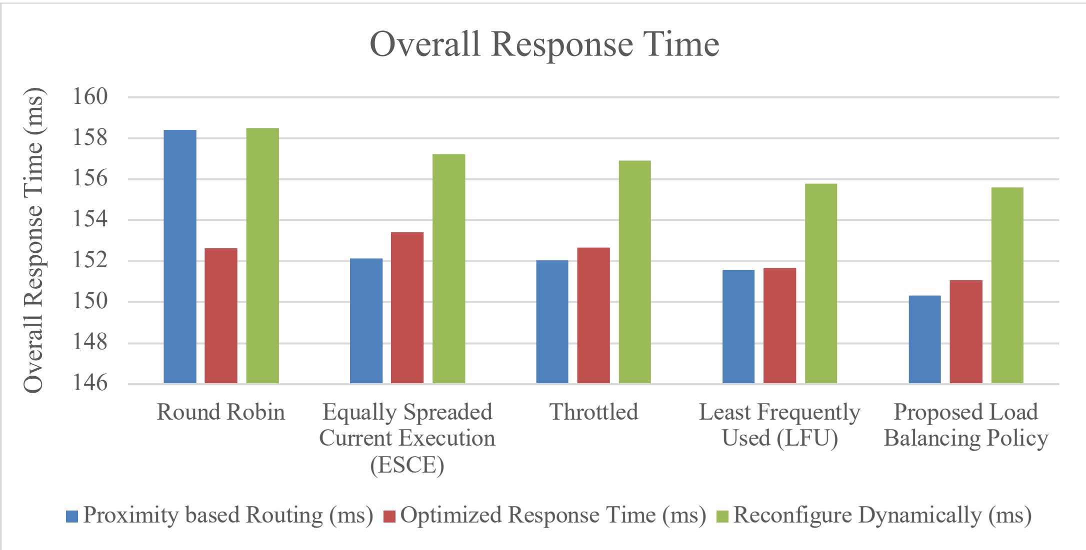


Figure 2: Overall Response Time Comparison

Table 2: Comparative Analysis of Data Center Processing Time

	Proximity based Routing (ms)	Optimized Response Time (ms)	Reconfigure Dynamically (ms)
Round Robin	0.96	1.44	7.02
Equally Spreaded Current Execution (ESCE)	0.59	2.21	5.75
Throttled	0.43	1.46	5.43
Least Frequently Used (LFU)	0.41	1.23	5.36
Proposed Load Balancing Policy	0.35	1.01	5.23

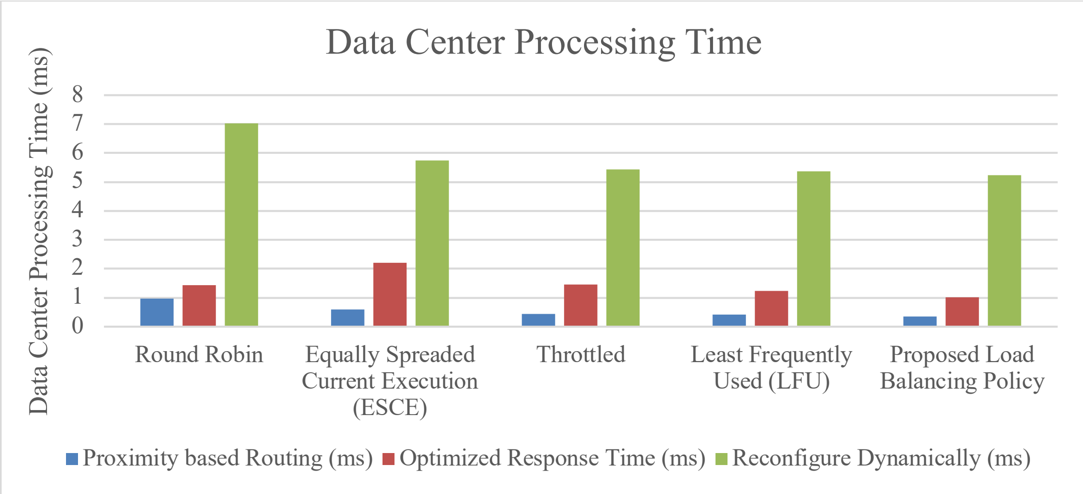


Figure 3: Data Center Processing Time Comparison

B. Load-Aware VM Placement and Overloaded / Underloaded Host Selection Methods

Algorithm 2: Load-Aware VM Placement

Input: hostsList, vmList
Output: LoadAwarePlacement of VMs

```
vmList.sortDecreasingUtilization()
Foreach vm in vmList do
  bestPlacement ← Min
  bestHost ← Null
    Foreach host in hostList do
      If host has enough resources for vm then
        Utilization ← estimateUtilization (host,vm) ➔
      Uhost,vm = Uhost + Uvm
        LoadIncrement ← estimate LoadIncrement (host,vm) ➔
      Loadhost,vm = Loadhost,vm – Loadhost
        placement ← utilization – LoadIncrement
     If placement > bestPlacement then
        bestHost ← host
        bestPlacement ← placement
     If bestHost != Null then
        LoadAwarePlacement.add (vm,bestHost)
return LoadAwarePlacement
```

Algorithm 3: Overloaded / Underloaded Host Selection

Input: hosts
Output: overloaded host detection & underloaded host detection

```
isHost.underloaded = False
isHost.overloaded = False

If Host.Utilization > 85%
     is.Host.overloaded = True
End If
Else if Host.Utilization < 20%
      isHost.underloaded = True
End If
```

Experimental Setup for Load-Aware VM placement and Overloaded / Underloaded Host Selection methods:

Overloaded / Underloaded host selection and Load-Aware VM placement methods have been implemented on OpenStack. To carry on research experiment on an extensive level and to handle huge workload, OpenStack has been hosted over real-time AWS server. For AWS infrastructure, a t3.2xlarge sized dedicated EC2 instance has been deployed with the capacity of 8 vCPU, 32 GiB memory, and 4 Gbps network performance. We have deployed around 12000 VMs dynamically from the set of 1000 available hosts considering that a single VM may migrate multiple times until completes its execution of the provided workload. On the other end, we have fed predefined sample dataset as workload from the Kaggle. Usage of large dataset resulted in efficient measurement of system scalability and performance.

Load-Aware VM placement and Overloaded / Underloaded host selection methods have been implemented along with well-known methods Threshold Minimum Migration Time (ThrMmt), Threshold Random Selection (ThrRs), Median Absolute Deviation Minimum Utilization (MadMu), and Median Absolute Deviation Random Selection (MadRs). Results analysis are done with three criteria, Energy consumption, No. of VM migration and Mean time value. The results are described in Table 3. Figures 4, 5 and 6 are representing results in bar charts for energy consumption, number of VM migration, and mean time analysis respectively.

Table 3: Comparison of Proposed vs Traditional approaches

Criteria	Approach				
	ThrMmt	**ThrRs**	**MadMu**	**MadRs**	**Proposed**
Energy Consumption (KW)	191.73	189.36	193.42	188.24	185.66
No of VM Migration	26583	26379	26963	26138	25904
Mean time before VM migration (ms)	19.33	19.71	19.67	19.71	19.77

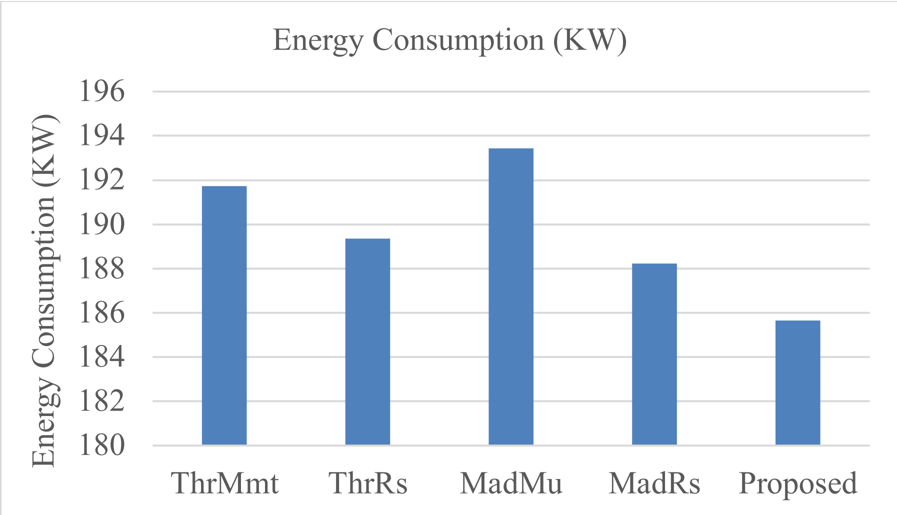


Figure 4: Energy Consumption Comparison

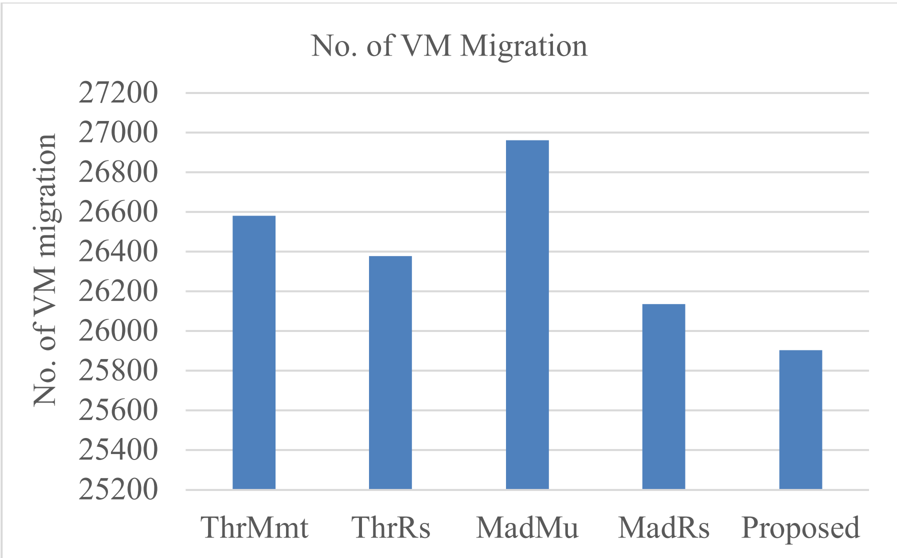


Figure 5: Number of VM Migration Comparison

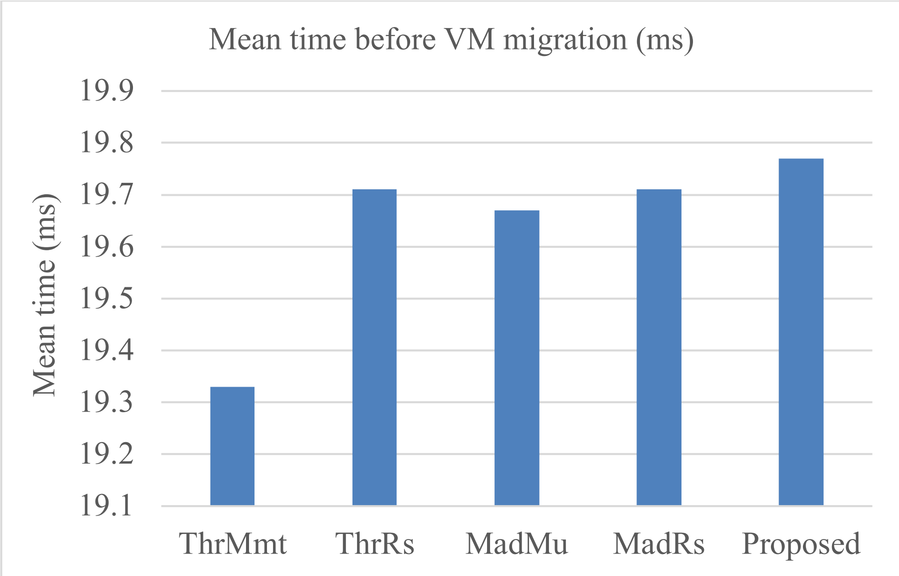


Figure 6: Mean Time Analysis before VM Migration

7. Achievements with respect to objectives

In all existing traditional approaches mostly either Mips or Memory has been considered to improve system performance and to reduce energy consumption. In proposed methods, Task submission and VM placement, Load-Aware VM placement, and Overloaded / Underloaded host selection average utilization of Memory, Mips, and Bandwidth have been used. That contributes significant improvements in the results. Task submission and VM placement method considerably improve performance on overall response time and data center processing time compared to traditional Round Robin and other algorithms. Overloaded / Underloaded host selection method mostly focuses on VM migration strategy while the Load-Aware VM placement method helps to reduce energy consumption.

The proposed approach has 4% less VM migration and 3% less energy consumption compared to the MadMu method, while it has 2.28% improvements for Mean time compared to ThrMmt method. Task queues are managed by proposed algorithms that confirm better Quality of Service (QoS) and SLA management.

8. Conclusion

There is a necessity to build a huge number of cloud data centers because of the drastic increase in internet users that in turn leads to avail high demand for computing capacity. A hugh amount of energy is required to operate and manage data centers. The proposed approach of Task submission and VM placement, Load-Aware VM placement, and Overloaded / Underloaded host selection focused on average load capacity, which is calculated as an average utilization of Memory, Mips and Bandwidth. Task submission and VM Placement method is effective for overall response time and data center processing time. Along with Load-Aware VM placement, the selection of overloaded / underloaded hosts algorithm did not allow users to allocate additional workload. Also, they minimized the number of VM migration and improved mean time value. All in all, they helped reduce energy consumption and optimized system performance.

Future Work

The proposed approach with three unique methods has performed well with comparatively better results. Total load, utilization of host, and utilization of VM are calculated with Memory, Mips and Bandwidth. This calculation might be explored further with additional parameters.

In proposed methods, the input workload tasklist has been assigned in a first come first serve manner. Proposed system can be explored further with priority-based task execution with new workload assignment strategy.

9. Copies of papers published and a list of all publications arising from the thesis

The publication detail for the work is as below:

[1] Hitesh A. Bheda, Chirag S. Thaker and Darshan B. Choksi, “Performance Enhancement and Reduce Energy Consumption with Load Balancing Strategy in Green Cloud Computing”, Progress in Advanced Computing and Intelligent Engineering. Advances in Intelligent Systems and Computing, Springer, Singapore, Vol 1299, pp. 585-597, 2021. Print ISBN: 978-981-33-4298-9; Online ISBN: 978-981-33-4299-6; https://doi.org/10.1007/978-981-33-4299-6_48

[2] Hitesh Bheda, Chirag Thaker and Sanjay Shah, “An Optimized VM Placement Approach to Reduce Energy Consumption in Green Cloud Computing”, In Proceedings of ACM DSMLAI conference (2020 DSMLAI), August 9-12, 2021, Windhoek, Namibia. ACM, New York, NY, USA. ACM ISBN 978-1-4503-8763-7/20/06; https://doi.org/10.1145/3484824.3484894 [Paper accepted, In the production]

[3] Hitesh A. Bheda, Chirag S. Thaker, Sanjay M. Shah, Darshan B. Choksi, “VM Relocation based Energy Reduction in Green Cloud Computing”, Journal of Xidian University, Vol. 14, Issue 7, pp. 74-78, 2020. ISSN: 1001-2400

10. References

[1] The NIST Definition of Cloud Computing, 2011. https://www.nist.gov/news-events/news/2011/10/final-version-nist-cloud-computing-definition-published#:~:text=AccordingtotheofficialNIST,andreleasedwithminimalmanagement

[2] Arman Shehabi, Sarah Smith, Dale Sartor, Richard Brown, Magnus Herrlin, Jonathan Koomey, Eric Masanet, Nathaniel Horner, Inês Azevedo, William Lintner. 2016. United states data center energy usage report.

[3] Chintan Doshi, Gaurav Verma, and k. Chandrasekaran, “A green mechanism design approach to automate resource procurement in cloud”, Procedia Comput. Sci., Vol. 54, pp. 108–117, 2015.

[4] Anton Beloglazov, Rajkumar Buyya, Young Choon Lee, Albert Zomaya, “A Taxonomy and Survey of Energy-Efficient Data Centers and Cloud Computing Systems”, Advances in Computers, Vol. 82, pp. 47–111, 2011. doi: 10.1016/B978-0-12-385512-1.00003-7.

[5] David Aikema, Andrey Mirtchovski, Cameron Kiddle, Rob Simmonds, "Green cloud VM migration: Power use analysis", 2012 International Green Computing Conference (IGCC), San Jose, CA, pp. 1-6, 2012. doi: 10.1109/IGCC.2012.6322249.

[6] Minu Bala, Devanand, “Virtual Machine Migration: A Green Computing Approach in Cloud Data Centers”, In: Satapathy S., Bhatt Y., Joshi A., Mishra D. (eds) Proceedings of the International Congress on Information and Communication Technology. Advances in Intelligent Systems and Computing, Springer, Singapore, Vol 439, 2016.

[7] K. Zhang, T. Wu, S. Chen, L. Cai and C. Peng, “A New Energy Efficient VM Scheduling Algorithm for Cloud Computing Based on Dynamic Programming”, 2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud), New York, NY, USA, pp. 249-254, 2017.

[8] Junaid Shuja, Raja Wasim Ahmad, Abdullah Gani, Abdelmuttlib Ibrahim Abdalla Ahmed, Aisha Siddiqa, Kashif Nisar, Samee U. Khan, Albert Y. Zomaya, “Greening emerging IT technologies: techniques and practices”, Journal of Internet Services and Applications, Vol. 8, No. 1, pp. 1–11, 2017. doi: 10.1186/S13174-017-0060-5.

[9] Ali Hammadi, Lotfi Mhamdi, “A survey on architectures and energy efficiency in Data Center Networks”, Computer Communications, Vol. 40, pp. 1–21, 2014. doi: 10.1016/J.COMCOM.2013.11.005.

[10] Yacine Kessaci, Melab Nouredine, El-Ghazali Talbi, “A multi-start local search heuristic for an energy efficient VMs assignment on top of the OpenNebula cloud manager”, Future Generation Computer Systems, Elsevier, Vol. 36, pp. 237–256, 2013. doi: 10.1016/J.FUTURE.2013.07.007.

[11] Yonghong Luo, Shuren Zhou, “Power Consumption Optimization Strategy of Cloud Workflow Scheduling Based on SLA”, WSEAS Transactions on Systems, Vol. 13, pp. 368-377, 2014.

[12] Huankai Chen, Frank Z. Wang, N. Helian, Gbola Akanmu, “User-Priority Guided Min-Min Scheduling Algorithm For Load Balancing in Cloud Computing”, 2013 National Conference on Parallel Computing Technologies (PARCOMPTECH), pp. 1-8, 2013.

[13] Veerawali Behal, Anil Kumar, “Cloud Computing: Performance Analysis of Load Balancing Algorithms in Cloud Heterogeneous Environment”, 2014 5th International Conference- Confluence The Next Generation Information Technology Summit, 2014.

[14] Kunjal Garala, Namrata Goswami, “A Performance Analysis of Load Balancing Algorithms in Cloud Environment”, IEEE, International Conference on Computer Communication and Informatics, 2015.

[15] B. Kruekaew, W. Kimpan, “Virtual Machine Scheduling Management on Cloud Computing Using Artificial Bee Colony” Proceedings of the International MultiConference of Engineers and Computer Scientists 2014, Hong Kong, Vol I, 2014.

[16] Wang, Lin, Zhang, Fa, Aroca, Jordi Arjona, Vasilakos, Athanasios V., Zheng, Kai,

Hou, Chenying, Li, Dan, Liu, Zhiyong, "GreenDCN: A General Framework for Achieving Energy Efficiency in Data Center Networks", IEEE Journal on Selected Areas in Communications, Vol. 32, Issue 1, pp. 4–15, 2014. Doi: 10.1109/JSAC.2014.140102.

[17] Akindele A. Bankole, Samuel Adesoye Ajila, "Cloud client prediction models for cloud resource provisioning in a multitier web application environment", Proceedings - 2013 IEEE 7th International Symposium on Service-Oriented System Engineering, SOSE 2013, pp. 156–161, 2013. doi: 10.1109/SOSE.2013.40.

[18] Yunliang Chen, Xiadodao Chen, Wangyang Liu, Yuchen Zhou, Albert Zomaya, Rajiv Ranjan, Shiyan Hu, "Stochastic scheduling for variation-aware virtual machine placement in a cloud computing CPS" Future Generation Computer Systems, 2017. doi:10.1016/j.future.2017.09.024.

[19] Sourav Kanti Addya, Anurag Satpathy, "A Game Theoretic Approach to Estimate Fair Cost of VM Placement in Cloud Data Center", IEEE Systems Journal, 2017. DOI 10.1109/JSYST.2017.2776117.