2023 3rd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE) | 979-8-3503-9926-4/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICACITE57410.2023.10182878

# Optimization of Load Balancing Algorithm in Cloud Computing

Ranjan Walia
*UCRD, Chandigarh University*
Mohali, Punjab, India
ranjanwalia@gmail.com

Lavish Kansal
*Lovely Professional University*,
Phagwara, Punjab, India
lavishkansal33lpu@gmail.com

Mahipal Singh
*Uttaranchal Institute of Technology, Uttaranchal University*
Dehradun, Uttarakhand, India
deansoa@uttaranchaluniversity.ac.in

Komuravelly Sudheer Kumar
*School of Computer Science and Artificial Intelligence, SR University*
Warangal, Telangana, India
k.sudheerkumar@sru.edu.in

R. M. Mastan Shareef
*Department of Science & Humanities, St. Marin's Engineering College*
Secunderabad Telangana, India
mastan.shareef786@gmail.com

Suresh Talwar
*Department of IT, St. Martin's Engineering College*,
Secunderabad Telangana, India
sureshtalwarit@smec.ac.in

***Abstract:*** **A rapidly expanding service, cloud computing (CC) uses a pay-per-use business model. As far as capacity, organization, web administrations, and so forth, innovation offers various administrations. Regardless, the expansion of these organizations and the enormous flood in user demand has made it trying to stay aware of execution as per QoS assessment and SLA chronicles that cloud suppliers make available to businesses. This growth brought about difficulties like load balancing. In addition, it became challenging to meet customer expectations for response speed and work scheduling deadlines. This research suggests an optimal approach based on schedule limitations using the Machine Learning Classification technique to overcome these issues. The main goals of the suggested technique are to increase productivity, optimise server resources by taking into account the importance of various users' tasks, and prevent server failure. Based on the most recent literature, our suggested method will address the aforementioned problems and the existing research gap.**

***Keywords: Optimization, Load balancing, Cloud computing, machine learning, virtual machine.***

## I. INTRODUCTION

An arising innovation called cloud computing (CC) offers services for facilitating and getting documents and data remotely as opposed to locally on a PC. a concept put forth by Prof. Ramnath Chellapa in 1997 [1] and characterised [2] as a powerful framework engineering meant to offer clients a wide range of useful services over the internet. The technology aims to enhance global trade thanks to its scalable environment and affordable hardware. The three delivery techniques that CC employs are Platform as a Service, Software as a Service, and Infrastructure as a Service. Web browsers are used to access SaaS services like Google Docs, Gmail, and others. By offering platforms and programming languages, cloud suppliers additionally aid the improvement of client services; this is known as PaaS. Clients have more control over the operating system and less influence over the operating system in data centers, the final version of IaaS for data storage. Large IT firms like Google and Microsoft offer pay-per-use services. SaaS is the most popular service model among the three that enterprises utilize, according to research [3], and this is because it is the simplest service that is easily accessed using web browsers and doesn't require installation.

Over time, cloud services have expanded incredibly. According to data from 451 Research, 60% of the workload for firms is being done in the cloud in 2019, compared to 45% in 2018 [4]. This information demonstrates that business use of cloud services has increased significantly. Such service expansion frequently presents difficulties for cloud service providers in maintaining the caliber of services offered to their customers. One of the three vital difficulties of CC is execution, as issues like load balancing may make CC applications perform ineffectively, which will influence client satisfaction [5].

Rather than different innovations like grid computing and utility processing, virtualization is a pivotal part of CC applications [1]. It makes virtual holders known as Virtual Machines from actual components like operating systems, servers, storage devices, etc (VMs). Virtualization, in other terms, adds an amorphous barrier between software and hardware [6]. The ability to operate numerous VMs on a single hardware layer is achieved with the aid of hypervisors, also known as Virtual Machine Monitor (VMM). There are two types of VMM, as shown in Fig. 1 below; type 1 works directly with the hardware layer, but type 2 requires a host operating framework to provide help, stockpiling, and other capabilities. Each VM has two layers: the application layer and the guest OS (Windows, Linux, or Mac). This capability enables CC to provide clients with more scalable on-demand services. Since CC heavily relies on virtualization technology, poor virtual machine migration and task allocation can have a significant impact on how well applications run.

Future customer requests are typically made in the form of VMs in a cloud environment, and suppliers utilizing the IaaS model should ensure the nature of their administrations so that users' tasks are finished in the apportioned time [8]. To provide a correct and balanced workload across servers, client demands are circulated through the data merchant to the suitable VM in light of a scheduling system. This can be accomplished by offering a reliable load-balancing method. In CC, load balancing aims to achieve two main objectives: to start with, asset portion, which involves relegating undertakings to the fitting VMs to such an extent that no VM is exhausted or has next to no burden. Second, following the

assignments, tasks are planned to be accomplished by a specific deadline and in compliance with user needs.

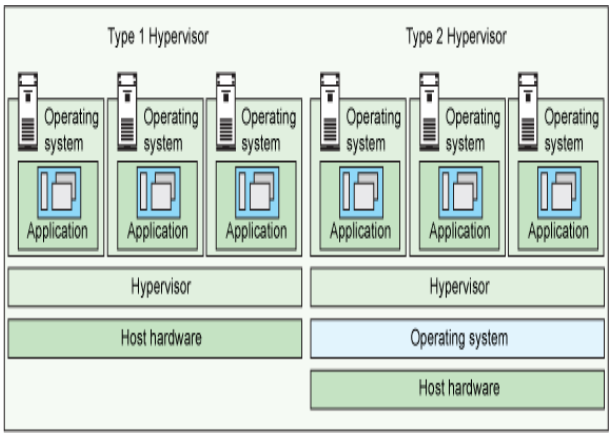


Fig. 1. Hypervisor types [7]

Round Robin (RR) and other existing load balancing methods are static calculations and should not yet be used in special circumstances, like CC, when traffic varies dramatically [9]. Similar to this, scheduling algorithms like First Come First Serve (FCFS) that schedule activities based on arrival time ignore essential SLA requirements like task deadlines and do not take task priority into account, which tends to slow down response times for Supervised and Unsupervised Learning are the two main categories for machine learning. The example of the contribution to the dataset is equivocal and unlabeled in unsupervised learning, in this manner the model is prepared to have the option to figure out how to sort out the information with the least possible mistakes. Conversely, the information test in directed learning is marked with a particular result. Both linear and non-linear issues, similar to grouping, can be settled utilizing this sort of preparation. A decision-making issue including characterization is one in which an item should be arranged into the fitting foreordained class given the number of characteristics that characterize or recognize its basic tasks [10].

Because of the rising development in the utilization of cloud services, the tasks should be done accurately to amplify client bliss. Different solicitations are being recorded from different clients in different regions. High-priority tasks must be completed first, and quick responses must be taken into account. Because of the rising interest in services, it is important to consider dispersing the responsibility similarly among clients and servers to further develop asset accessibility and effectiveness. At the point when assets are utilized actually, servers in server farms are optimized and overloading circumstances ought to be kept away from. Task migration, for example, is the most common way of moving an errand to a VM with a decreased load when a VM is overloaded with client demands. Task prioritisation, load balancing, and task migration all help CC applications run more efficiently.

The scope of this study is constrained to emphasize the need of giving assignments a higher priority based on a deadline criterion. The load-balancing techniques now in use have several drawbacks. The following categories describe the limitations: Some researchers frequently neglect the importance of task priority as a scheduling component [9] [11] [12]. Even though the need is expected, there is as yet a limitation that prevents jobs with equivalent need values from being thought about in the wake of planning finished [13]. Even though load balancing has been improved, the problem of task migration where jobs are assigned to VMs despite their overload has not yet been handled [14] – [16].

The suggested algorithm thus incorporates the Machine Learning Classification technique to address the aforementioned problems while also taking into account the following two factors: Undertakings are focused on in light of the cut-off time parameter b task movement when a new solicitation is given to a VM that is now overloading.

## II. Literature Review

The literature related to load balancing and AI research conducted by various analysts is reviewed in this part. Fig. 2 below provides an illustration of the scientific classification of load balancing and work scheduling algorithms to aid readers in comprehending and categorising the basic approach applied in the examination. The performance of CC applications can now be improved by strengthening the load-balancing strategy; however, there are still several limits that are highlighted in the literature listed below.

Researchers offer a dynamic load-balancing algorithm in [11]. Although static algorithms like RR are easy to construct, they are not appropriate for environments like CC where load varies regularly. The strategy decides the load on all VMs first and afterward stores the worth in a line. When a data center is unable to manage a heavy burden from an anticipated request, the elasticity notion is used. Otherwise, it will evaluate each VM's status and decide whether to move a task that is too busy to one suitable VM over to another. Assignments are made in the order of first come, first served, and the length of the task is determined at random. Although the makespan is decreased and the algorithm may operate well in real-time contexts, the scheduling strategy is centered on appearance time as opposed to task need.

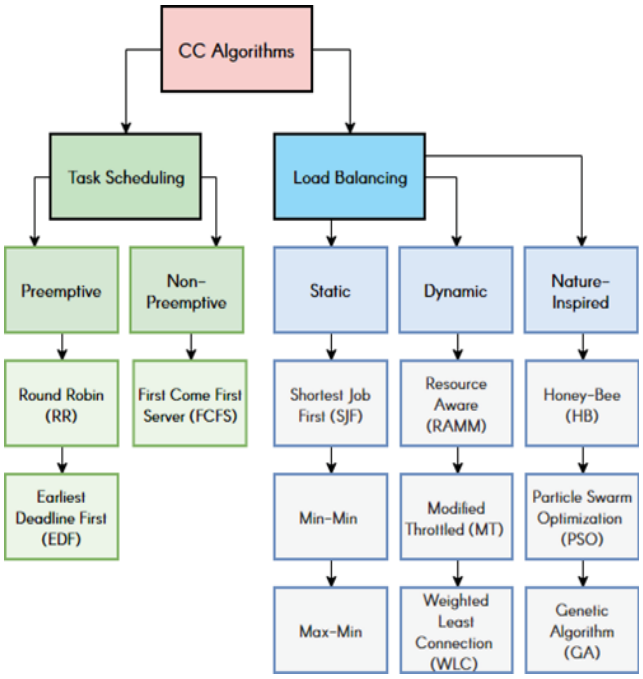


Fig. 2. Cloud computing algorithm taxonomy

The conventional Min-Min (MM) algorithm is improved by the resource-aware (RAMM) strategy, which is detailed in [12]. The inability to perform many tasks at once and

starvation for larger jobs, which results in an unbalanced VM load, are only a few of MM's drawbacks. Tasks can be assigned to resources (VMs) using the suggested algorithm, which maintains a matrix of these resources together with their calculated execution times (CTs) and completion times (ETs), respectively. The minimal CT and the projected execution time on the virtual machine are taken into account when scheduling tasks. Although the method shortens job completion times, it ignores VM allocation capacity and does not enforce task priority.

The method put forward by researchers in [13] takes job priority into account. It evaluates jobs of a similar length and picks the work with the most noteworthy need in light of its completion date. As a result, projects with shorter deadlines and shorter lengths are given to VM first. Even while the method shortens response times overall when compared to RR, it still ignores the possibility that two or more jobs have the same deadline, necessitating further improvement to priority [9] recommended a load balancing procedure where task dispersion is relying upon the present place of job count. In addition to minimizing overloaded scenarios, VM priorities are taken into account. Similarly to this, when a job is assigned, VMs are updated and other tasks are notified. For the sequencing of tasks entering the holding-up line, FCFS is utilized. The undertaking should be relegated to a VM with minimal measure of interest if not it will be deferred until an accessible VM is found. Even though academics have of late utilized clustering algorithms and order AI to force need, there is still potential for improvement as can be seen in the literature listed below.

The authors of [17] used Mean Shift Clustering Algorithms and Dominant Sequence Clustering (DSC) to address load balancing and work scheduling issues. It evaluates projects according to two criteria: deadline and makespan. User tasks are organized by their importance. Then, utilizing kernel functions, the MSC technique is employed to cluster virtual machines (VM). For task appropriation, the server with the least weight and association is picked. The WLC method, then again, utilizes designated undertakings to compute server load, which is oftentimes erroneous because the load changes each time a work is dispensed or redistributed. Also, when the weight of every server is laid out, it is more challenging to in a split second change it [18]. Although this method lengthens the response time, further server resource optimization is still required.

Based on information from numerous log files, authors in [14] categorised users' tasks and virtual machines (VMs). The benefit of using log records is to understand how online users behave and, specifically, to obtain precise project size information. The classification of tasks into three categories Light, Medium, and Heavy, and the calculation of their resource requirements results in more effective task allocation and greater resource utilization. After classifying the jobs, they will calculate the CPU and RAM consumption (from 0-100%) of the VMs (how many tasks can be deployed) to estimate the capacity. This data is utilized to order virtual machines into 5 gatherings: idle (1%), light, ordinary, semi-weighty, and weighty/over-load (>90%). Then, depending on their sizes, heavy assignments are assigned to sit or light groups, while light assignments will be sent off to heavy or normal groups because they don't need a lot of resources. This approach has various drawbacks, including the fact that tasks can still be assigned to VMs even when they are overloaded and that there is significant delay since task length is not taken into account.

A prediction method for VM availability. It uses a computed number to rank tasks according to priority, which is the amount of four elements with fixed loads. VMs are categorized based on their MIPS rating and transmission capacity [19]. The task is given to the appropriate VM with the least amount of execution time possible; despite this, the task's completion time is anticipated and the VM's accessibility is established. This system has advanced thanks to authors in [20] who used Global and Local Queues. The two kinds of upcoming jobs are computational (rapid processing) and communication (high MIPS). Due to the use of non-pre-emptive techniques, there are still some scheduling restrictions.

The task classification using supervised ML is described in [21]. The same task criteria mentioned in [16] have been employed by researchers to enforce priority. The learning algorithm uses a priority queue technique and sorts tasks according to the deadline. The data set is implemented by randomly assigning values to tasks. With fluctuating quantities of jobs, the authors used 22 different classifier models, (for example, straight SVM, basic trees, and so on.). As per the outcomes, Boosted Tree (400 undertakings) has the most elevated exactness rate, coming in at 94.3%. The strategy diminishes planning reaction time yet disregards task relocation.

In this work [22], analysts utilized lines to address the need issue in light of three rules: task length, task age, and task deadline. The undertakings with the most elevated need will be planned first when different needs are characterized. By moving the most reduced need errands to the front of the line, the strategy can eliminate holding up times, however, it disregards task movement starting with one virtual machine and then onto the next. In this way, the methodology in all actuality does effectively address load-balancing issues.

## III. Methodology

A novel strategy is recommended to work on the exhibition of CC applications to resolve the issues recorded in the issue depiction segment A and ML arrangement and B. Proposed Scheduling Algorithm are the two fundamental divisions of the archive.

### *A. ML Classification*

The main thing seen is an open-source informational set, perhaps comprising weblogs. The information collection yields some errand points of interest, like the task ID, length, and appearance time. To set up the preparation data sets with names for each property, such information is necessary. However, as the majority of online data sets do not include information regarding deadlines, projects can be given deadlines at random [23].

In the review, it was found that the Supported tree is the most appropriate model for need scheduling out of all the ML classifiers. Since the unique climate of CC requires the work of countless exercises, this classifier can plan 400 tasks with a 97% precision rate. According to their date and work type (length), occupations have been divided into three need bunches using this method: Light (2000 bytes), Heavy (>4800 and 8000 bytes), and Medium (>2000 and 4800 bytes) [6].

Another sorted data set that will be utilized to a limited extent B is the outcome.

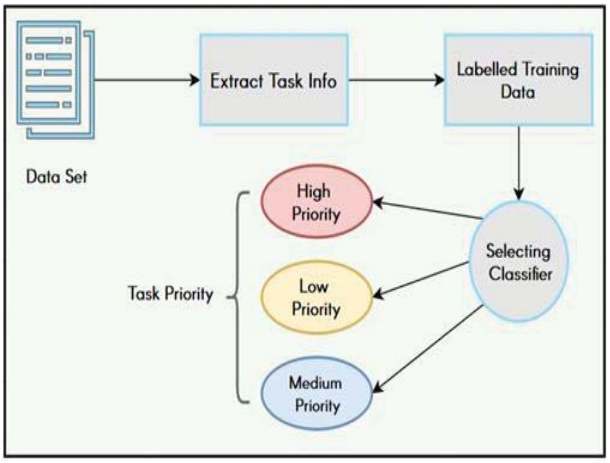


Fig. 3. Classification Process

*B. Scheduling algorithms Under Discussion*

The algorithm pseudo-code used for job scheduling to VM is described in full below:

**Algorithm:** Priority based classification ML and deadline constraint for better load balancing

Input: List of Classified Tasks (t1,t2,t3,…,tn) & VM
**Output:** Scheduled tasks (task mapped to VM)
**Begin**
**Initialization:** DT; AT; TT; MIPS; BW.
1. Sort tasks according to DT ascendingly
**If** two tasks have same DT then
Pick task with earliest AT
**Else**
Priorities based on (DT & TT)
**For each** VM
2. Compute Utilization (MIPS & BW)
3. Sort VMs according to their Utilization (%)
**Repeat**
**If** VM is available & task allocated to heavy VM
**then**
Migrate task to less utilized VM
**Else**
Start scheduling
**Until** all tasks allocated to a VM
**End**

The scheduling algorithm's task migration and priority enforcement produce a balanced workload. The three primary task parameters are used to prioritize the classified tasks:

a) Task Deadline Time (DT): This time limit specifies how long a task has to complete its goal, and it is expressed in milliseconds.

b) Task Type (TT): according to their length, divides tasks into three major categories: Medium, Light, and Heavy.

c) Task Arrival Time (AT): This time stamp, which is expressed in milliseconds, indicates when jobs are added to the ready queue (ms).

A task's importance is determined by how little work it requires and how soon it must be completed. The task with the earliest arrival time is chosen if two tasks have the same deadline value.

The two primary parameters that will be used to calculate VM usage (load) are:

a) A measure of the CPU component that shows how quickly instructions are executed is called Millions of Instructions per Second (MIPS).

b) Bandwidth (BW): This term refers to VM capacity. Megabytes per second are used to measure it (MBPS).

## IV. Results And Discussion

Sections A and B are combined in our final arrangement, which is a hybrid model that considers both job scheduling and load balancing.

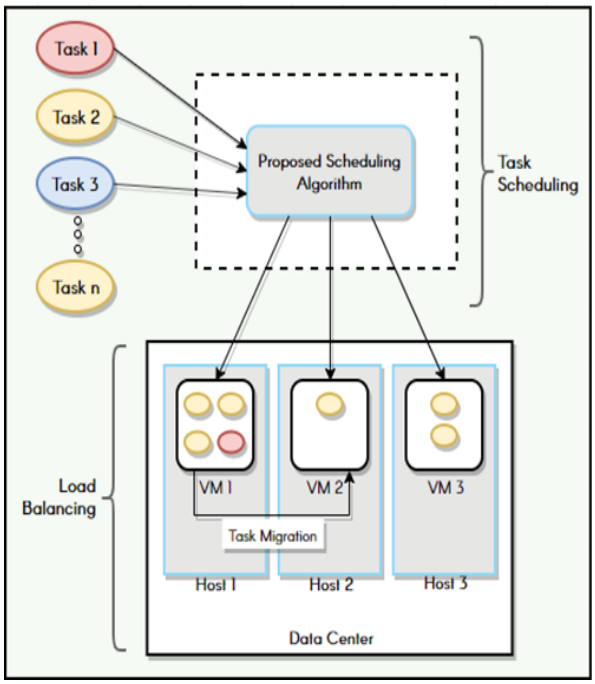


Fig. 4. Environment for Cloud Computing Based on Models

To provide customers with high-quality services like storage, deployment, web services, etc., cloud computing is a quickly growing industry. The demand for high performance will continually rise as a result of its expanding market. Load balancing, which has a strong connection to SLA, is among the most crucial concepts. To assist with need and task migrations while taking the cut-off points learned from fresh literature and the writer's knowledge into account, this research proposes a model-based ML depiction. Furthermore, we offered scientific classification to recognize different algorithms as indicated by their scheduling strategy and climate. To highlight the research gap in this work, an analysis of recent methods for working on the display of CC apps is also provided.

## V. Conclusion

To effectively distribute work to virtual machines (VMs) and prevent VM overloading problems, this study intends to propose a component that may be used in cloud computing

(CC). Additionally, when applied to new activities entering the server center, machine learning (ML) categorization methods can preprocess data to produce better and faster outcomes. The adoption of a good classifier like Supported Tree can result in an AI model with high accuracy and little training time. More task-related parameters will be taken into account in subsequent attempts to increase user happiness, and different AI classifiers will be evaluated continuously to raise the success rate of processing the dataset. Overall, this work emphasizes the potential advantages of applying ML classification techniques to CC to better distribute tasks and increase system effectiveness. The adoption of such technologies is becoming more crucial due to the rapid development of cloud computing and the rising demand for effective task management. The precision and scalability of these techniques can be enhanced with more research in this field, which will ultimately result in cloud computing systems that are more effective and efficient.

## REFERENCES


[1] M. Agarwal and G. M. S. Srivastava, "Cloud computing: A paradigm shift in the way of computing," Int. J. Mod. Educ. Comput. Sci., vol. 9, no. 12, pp. 38-45, 2017.

[2] M. Nazir, "Cloud computing: overview & current research challenges," IOSR J. Comput. Eng., vol. 8, no. 1, pp. 14-22, 2012.

[3] B. El Zoghbi and C. Chedrawi, "Cloud Computing and the New Role of IT Service Providers in Lebanon: A Service-Dominant Logic Approach," in ICT for an Inclusive World: Industry 4.0–Towards the Smart Enterprise, 2020, pp. 425-437.

[4] D. A. Shafiq, N. Z. Jhanjhi, and A. Abdullah, "Proposing a load-balancing algorithm for the optimization of cloud computing applications," in 2019 13th Int. Conf. Math., Actuarial Sci., Comput. Sci. Stat. (MACS), 2019, pp. 1-6.

[5] N. Zanoon, "Toward Cloud Computing: Security and Performance," Int. J. Cloud Comput. Serv. Archit., vol. 5, no. 5/6, pp. 17-26, 2015.

[6] B. Singh and G. Singh, "A study on virtualization and hypervisor in cloud computing," Int. J. Comput. Sci. Mobile Appl., vol. 6, no. 1, pp. 17-22, 2018.

[7] B. P. Tholeti, "Hypervisors, virtualization, and the cloud: Learn about hypervisors, system virtualization, and how it works in a cloud environment," IBM developerWorks, Sep. 2011. [Online]. Available: https://www.ibm.com/developerworks/cloud/library/cl-hypervisors-cloud/.

[8] M. Adhikari and T. Amgoth, "Heuristic-based load-balancing algorithm for IaaS cloud," Future Gener. Comput. Syst., vol. 81, pp. 156-165, 2018.

[9] W. Hashem, H. Nashaat, and R. Rizk, "Honey bee-based load balancing in cloud computing," KSII Trans. Internet Inf. Syst. (TIIS), vol. 11, no. 12, pp. 5694-5711, 2017.

[10] V. Panwar, D. K. Sharma, K. V. P. Kumar, A. Jain, and C. Thakar, "Experimental Investigations And Optimization Of Surface Roughness In Turning Of EN 36 Alloy Steel Using Response Surface Methodology And Genetic Algorithm," Mater. Today: Proc., vol. 46, pp. 1715-1722, 2021.

[11] M. Kumar and S. C. Sharma, "Dynamic load balancing algorithm for balancing the workload among virtual machines in cloud computing," Procedia Computer Science, vol. 115, pp. 322-329, 2017.

[12] S. A. Ali and M. Alam, "Resource-Aware Min-Min (RAMM) algorithm for resource allocation in a cloud computing environment," arXiv preprint arXiv:1803.00045, 2018.

[13] R. Aluri, S. Mehra, A. Sawant, P. Agrawal, and M. Sohani, "Priority-based Non-Preemptive Shortest Job First Resource Allocation Technique in Cloud Computing," Int. J. Comput. Eng. Technol. (IJCET), vol. 9, no. 2, pp. 132-139, 2018.

[14] M. Elrotub and A. Gherbi, "Virtual machine classification-based approach to enhanced workload balancing for cloud computing applications," Procedia Computer Science, vol. 130, pp. 683-688, 2018.

[15] U. R. Saxena, P. Sharma, and G. Gupta, “Comprehensive Study of Machine Learning Algorithms for Stock Market Prediction during COVID-19”, J. Comp. Mech. and Mgmt., vol. 1, no. 2, pp. 14–21, Dec. 2022.

[16] N. Er-Raji and F. Benabbou, "Priority task scheduling strategy for heterogeneous multi-datacenters in cloud computing," International Journal of Advanced Computer Science and Applications, vol. 8, no. 2, 2017.

[17] A. Al-Rahayfeh, S. Atiewi, A. Abuhussein, and M. Almiani, "A novel approach to task scheduling and load balancing using the dominant sequence clustering and mean shift clustering algorithms," Future Internet, vol. 11, no. 5, p. 109, 2019.

[18] D. Sharma, V. Kudva, V. Patil, A. Kudva, and R. S. Bhat, “A convolutional neural network based deep learning algorithm for identification of oral precancerous and cancerous lesion and differentiation from normal mucosa: A retrospective study,” Engineered Science, vol. 18, p. 278-287, 2022.

[19] S. Khurana and R. K. Singh, "Virtual machine categorization and enhance task scheduling framework in a cloud environment," in 2018 International Conference on Computing, Power and Communication Technologies (GUCON), pp. 391-394, IEEE, 2018.

[20] N. Miglani and G. Sharma, "An adaptive load balancing algorithm using categorization of tasks on a virtual machine based upon queuing policy in a cloud environment," Int J Grid Distrib Comput, vol. 11, no. 11, pp. 1-2, 2018.

[21] M. Shah, N. Naik, B. K. Somani and B. Z. Hameed, "Artificial intelligence (AI) in Urology-Current use and future directions: An iTRUE study," in Turkish Journal of Urology, vol. 46, no. Suppl 1, pp. S27-S32, Nov. 2020, doi: 10.5152/tud.2020.20291.

[22] K. Banerjee and S. Dua, “Astoundingly Smart System Furnishing Ranking of Big Data in Search Engines”, J. Comp. Mech. and Mgmt., vol. 1, no. 1, pp. 19–29, Sep. 2022.

[23] S. Deepa, “Metaheuristics for Multi Criteria Test Case Prioritization for Regression Testing”, J. Comp. Mech. and Mgmt., vol. 1, no. 1, pp. 42–51, Oct. 2022.