Enhancing Cloud Computing Efficiency: Fuzzy Based Task Classification for Better Resource Management

Mubarak Banisakher

Hillsborough Community College, United States,
mbanisakher@hccfl.edu

Marcie Yordy

Hillsborough Community College, United States,
myordy2@hccfl.edu

***Abstract* - This paper proposes a new task classification model for cloud computing to address inefficiencies that occur when the variability of the user task demand leads to suboptimal resource allocations. The paper studies the incoming tasks and classifies them, via a fuzzy clustering algorithm, into the base workload. Not only does it contribute to providing a fairer estimation of resource need for each task, but it also leads to increased use of resources and reduced cost of cloud maintenance. Experimental comparisons with the existing algorithms confirm that the model does indeed contribute to saving resources. It also categorizes the job into light, heavy, compute-intensive, and memory-intensive, which further optimizes the process of scheduling and allocation in the cloud computing environment.**

Key words - Cloud Computing, Utilization of Assets, Classification, Approximate Clustering Method, Buffered Data Streams

I. INTRODUCTION:

Cloud Computing is a new technology that could potentially revolutionize IT implementation and delivery [3]. Cloud computing is considered an innovative model for IT service sourcing that generates value for the adopting enterprises [25]. It enables enterprises to focus on their core business activities, and, as a result, productivity is increased [13]. The adoption of cloud computing is growing rapidly due to the scalability, flexibility, agility, and simplicity it offers to enterprises [13,27]. It offers several services presented in three models: Software as Service (SaaS), Platform as Service (PaaS), and Infrastructure as Service (IaaS). Software as Service (SaaS) provides applications or software to end-users, Platform as Service (PaaS) provides access to platforms, and Infrastructure as Service (IaaS) offers processing storage service [22].

The nature of application workloads ranges from regular and periodical up to even the unpredictable handed over to the cloud environment [1]. It is pertinent to understand their nature so that maximum utilization of resources is guaranteed and cost is minimized. This information is useful for cloud users who are discerning in their demands, and for providers in provisioning the necessary resources for application execution[2]. Methodologies for enabling energy-efficient dynamic task scheduling: a number of workload characterization methods and even more task-classification and merging methods are proposed. These techniques also focus on policies for VM (Virtual Machines) create, destroy, and migrate [3]. Comprehensive surveys are reviewed on workload characterization, issues on their targeted application domain, characterization approaches, and techniques applied [4]. There have been many works with respect to the analysis of workloads in Google compute clusters, in terms of resource utilization and server characteristics [5–7]. Techniques have been developed in the near past that assigns tasks to VMs with minimum workloads and thus enhances work balancing as well as performance accordingly. The multiqueue interlacing peak scheduling method (MIPSM) has been proposed for task scheduling that employs classification techniques which subdivided tasks according to their resource requirements [8].

New techniques have been developed that increase the degree of load balancing and work to ascertain the capability of each VM so as to maximize the use of resources [9]. An average resource utilization is used, considering tasks are classified considering size. But sometimes, misclassification of a new task can lead to performance degradation, like the task being assigned to an overloaded machine resulting in delays [10]. Traditional hard clustering techniques are poor in workload classification because of the unpredictable arrival patterns of workloads, which lead to inaccuracy in class assignment of work load, hence affecting energy consumption and resource utilization in cloud data centers. Few studies, however, have focused on classification of workloads in order to schedule, energy consumption and load balancing but poorly address resource utilization and VM creation [11].

There are notable benefits associated with an advanced knowledge of the characteristics of task workloads in order to avoid wastage of resources and to improve the quality of decisions related to the assignment of tasks. In this approach, clustering-based resource estimation categorizes tasks into groups with similar features to get appropriate clusters, which in turn lead to avoiding unnecessary VM creation and ensuring efficient task scheduling, balancing, and allocation [12]. This paper proposes a model, which constitutes two main constituents: the Task Resource Classifier (TRC) and the Task Buffer-Queue (TBQ). Task pattern with distinctive features is captured by the model using the Gustafson-Kessel Fuzzy C Means (GK-FCM) algorithm for classification [13]. Heavy, light, compute-intensive, and memory-intensive tasks are categorized first based on usage of CPU and memory, and then accuracy and saving on the resource ratio in the classification tasks. Contributions of this work lie in designing a framework for efficient classification of the incoming tasks based on resource demands to help in optimum VM allocation in the data centers, categorizing tasks in various types for creating appropriate VM, and utilizing knowledge of resource demands for better load balancing, which reduces downtime and improves the availability of services. This paper is organized in the context of related work, analysis of the dataset and task, the proposed methodology, experiments and results, and future work.

II. DATA SET EXAMINATION

A. Overview of Dataset:

The Alibaba cluster trace-v2017 provides information about the workload trace for 1300 machines in the duration of 12 hours [20]. This dataset includes the persistent online jobs, in addition to the batch jobs. It consists of six CSV (Comma Separated Values) files, especifically emphasizing on the duration and resource usage of the instances and tasks for the batch job running on the physical machines. For this research, the emphasis is only on batch jobs. In the dataset, every job is a composition of a lot of tasks, and at the lowest level, each of these tasks has lots of instances that are running operations.

B. Analysis of Tasks

First step is of the identification of the dimensions of the workloads. The batch workloads from the dataset has been focused in this study. Task and instance tables of the batch workload contain data about job-id, task-id, status, planned CPU and memory, actual CPU and memory, the occurrence of start, end times, and average CPU and memory. This analysis specifically targets those tasks that have been flagged as 'completed.' The other two important details in the resource utilization calculation is the used average CPU in cores, and the used average memory, normalized.

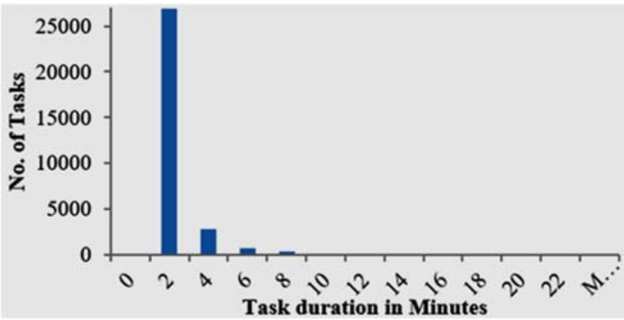

Figure 1. Task count and execution duration (minutes)

The results were that more than 80% of the tasks are of duration less than 5 minutes, while about 20% of them took more than 10 minutes and up to 2 hours, which one can categorize as short and long-duration tasks. CPU and memory are the two most influential characteristics of the task in terms of resource use. Equations (1) and (2) accordingly depict the average CPU and Memory occupied by task Tk, through focusing on the comprehensive resource utilized by each task.

$$mem\_T_k = average(\sum_{i=1}^{n}(real\_mem\_instance_i) \quad (1)$$

$$cpu\_T_k = average(\sum_{i=1}^{n}(real\_cpu\_instance_i)) \quad (2)$$

Figure 2 derives the insights by plotting the number of tasks against the normalized real average CPU usage of these tasks, where i represents values 1 to n, with n as the total number of instances in the k-th task. As shown in the plot in Figure 2, 70% of the tasks use fewer than 2 core CPUs, where just a tenth of the tasks need to use more than 32 core CPUs.

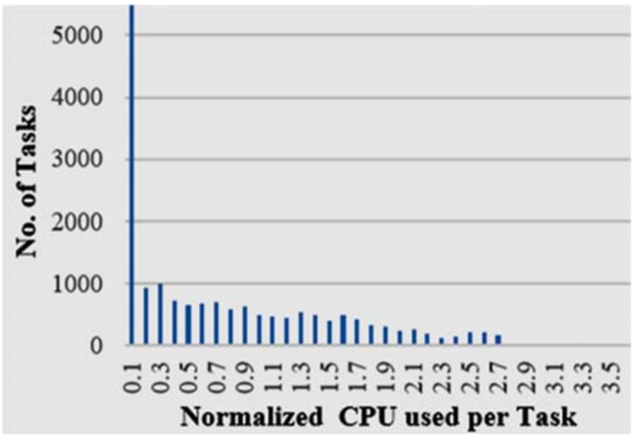

Figure 2. Normalized CPU used

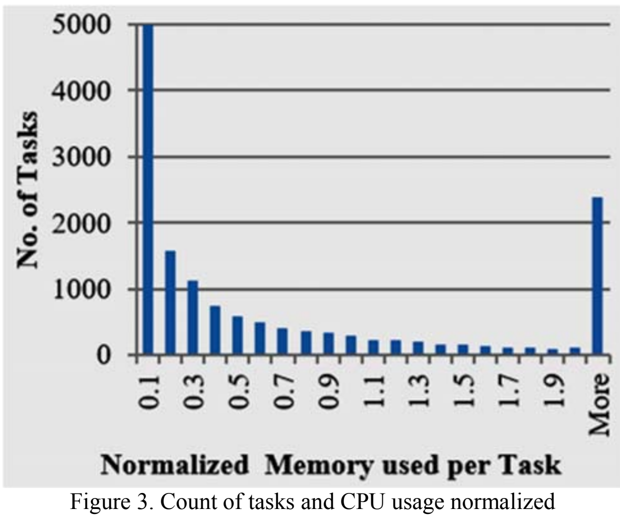

Figure 3. Count of tasks and CPU usage normalized

C. Resource Characterization

The variability in this respect is visible from the task completion times, as per Figures 1, 2, and 3 of this paper, and from the CPU and memory usage as well. One of the other important requirements of the task analysis presented in Section 3.2 is therefore workload characterization for the sake of efficient resource utilization. Resource categorization is proposed using Gustafson-Kessel Fuzzy C-Means (GK-FCM) for categorizing tasks on the basis of the resource consumed. Since the management of resources and cost is highly dependent on the CPU and memory, tasks are classified on the basis of usage of resources. Thus, task execution time along with CPU and memory utilizations are the key parameters in task clustering. Experiments are done over traces with Fuzzy clustering algorithms of Alibaba dataset. Post resource characterization analysis of the ttributes' short and long-range values are summarized in Table 1.

TABLE I. ATTRIBUTE SPECTRUM FOR BATCH TASKS

Attributes	Short Range	Long Range
Task Duration	0.0 to 300	301 to 8000
Normalized CPU	0.0 to 1.5	1.51 to 3.4
Normalized Memory	0.0 to 0.7	0.71 to 64

Table I outlines the categorization of tasks based on the analysis of their features and the defined ranges. Tasks with a duration of less than 300 seconds are classified as short-range, so, for the tasks that need more than 5 minutes of CPU time, which is like 300 seconds, we're calling those the long-range category. We're measuring CPU usage on this log scale that goes from 0 to 3.4. If a task uses less than 1.5 units of CPU, we're saying it's a light task. But if it's munching through more than 1.5 CPU units, then it's a heavy task.

III. METHODOLOGY

The methodology presented in this study focuses on classifying and grouping incoming tasks in a cloud environment, based on their varied resource requirements for execution. The architecture of the proposed method is depicted in Figure 4. In this setup, user applications are submitted to the cloud through an internet portal, where each application is broken down into a number of tasks. These tasks are characterized by a set of attributes, including execution time, status, CPU, memory, I/O, and bandwidth.

To train and test the proposed Task Resource Classifier (TRC) model, historical data is utilized. This data undergoes preprocessing before being applied to the TRC model. The TRC model employs a fuzzy clustering algorithm to categorize tasks into different classes. Based on three key attributes – task execution time, CPU, and memory – the incoming tasks are efficiently classified into eight distinct clusters. This approach enables a more nuanced and effective resource allocation in the cloud computing environment.

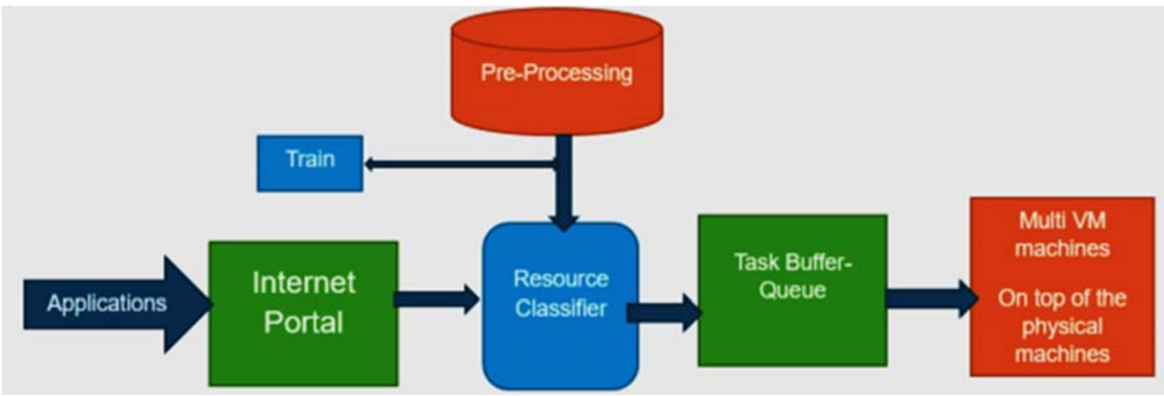

Figure. 4 Model for Classifying Tasks

The clustered tasks are then grouped into one of the four queues maintained by Task Buffer-Queue module (TBQ). Depending on the tasks' CPU and Memory requirement, clusters are merged into four buffer queues as light, heavy, compute intensive, and memory intensive.

A. Preliminary Data Processing

The pre-processing stage is crucial in handling the historical data, which may contain inconsistencies such as null values or incomplete job/task information. To prepare the data for experimentation, several cleaning and filtering steps are undertaken. Initially, tuples with null values are removed. The next step involves narrowing down the attributes to those specifically related to task execution time, CPU, and memory. These values are subsequently normalized to facilitate the application of the clustering algorithm.

B. Task Resource Classification Approach

In predicting resource needs, this study emphasizes task execution time, CPU, and memory utilization, as these factors significantly impact cost reduction and energy consumption in cloud environments. Referring to the resource characterization study and Table 1, the number of clusters was determined to be eight, labeled as C1, C2, C3,…, C8. These clusters are named based on resource usage, such as SSS, SSL, SLS, SLL, LSS, LSL, LLS, and LLL. The first letter represents the task duration as either long (L) or short (S), while the second and third letters denote CPU and memory usage, with S indicating light or small usage and L signifying large or heavy consumption. The set of workload clusters {C1, C2,…, C8} is derived from the GK-FSM clustering algorithm [23,24].

The primary goal of GKFCM clustering is the minimization of the objective function, ensuring accurate and efficient clustering.

$$J(V,\mu,F) = \sum_{i=1}^{c}\sum_{k=1}^{N}(\mu_{ik})^m\, d_{ik}^2 \quad (3)$$

In this context, k represents the data points, numbered from 1 to N, and i corresponds to the cluster count, ranging from 1 to c, which is predetermined. The variable n denotes the number of features for each data point. The fuzziness index, denoted as 'm', lies within the range of [1, ∞]. This index determines the extent to which the clusters may overlap. Within this framework, μik symbolizes the degree of membership, and dik represents the distance from the k-th data point to the i-th cluster center. The calculation of the membership degree is carried out as per Equation (4).

$$\mu_{ik} = \frac{1}{\sum_{j-1}^{c}\left(\frac{d_{ik}}{d_{ik}}\right)^{\frac{2}{(m-1)}}} \quad (4)$$

The term denoted as {μik}, it represents a c x N partition matrix, where c is the number of clusters and N is the number of data points. The distance dik, refers to the distance from the k-th data point to the i-th cluster:

:

$$d_{ik}^2 = \|x_k - v_i\|_{A_i}^2 = (x_k - v_i)^T A_i (x_k - v_i) \quad (5)$$

The center is computed as outlined in Equation (5). This calculation involves the squared inner-product distance norm, which is contingent on the symmetric matrix Ai and the cluster center vi.

In this framework, the variable i, ranging from 1 to c, is crucial in determining the shape and orientation of each cluster. The feature set of each data point is represented by xk, which is an array consisting of features xk1, xk2,…, xkn. These features collectively define the characteristics of each data point. The center of the i-th cluster is defined by vi, represented as an array [vi1, vi2,…, vin]. Here, V, denoted as {vi}, is a c x n prototype matrix, where c is the number of clusters and n is the number of features. The computation of this prototype matrix V is carried out according to Equation (6).

$$v_i = \frac{\sum_{k=1}^{N}(\mu_{ik})^m x_k}{\sum_{k=1}^{N}(\mu_{ik})^m} \quad (6)$$

$$A_i = [\rho_i \det(F_i)]^{1/n} F_i^{-1} \quad (7)$$

Where $i = 1, \ldots .\ k = 1, \ldots .\ N$

The GK-FCS (Gustafson-Kessel Fuzzy C-Means) algorithm employs an adaptive distance norm that is unique to each cluster, represented by Ai. This adaptive distance norm is integral to the algorithm's ability to effectively cluster data points. Ai is determined by estimating the data covariance, a process detailed in Equation (7). This estimation allows the algorithm to adapt the distance norm for each cluster based on the variance and distribution of the data points within it, enhancing the accuracy and efficiency of the clustering process.

In the context of the GK-FCS algorithm, ρ_i represents the volume of the i-th cluster. Alongside this, the fuzzy covariance matrix, denoted as F_i, plays a crucial role. The calculation of F_i is elaborated in Equation (8). The algorithm operates with fuzzy membership values, allowing every data point in the dataset to be partially associated with one or more clusters. This association is based on the membership values, with the key condition that the sum of a data point's membership across all clusters equals one. This ensures a proportional distribution of each data point across the clusters.

$$F_i = \sum_{k-1}^{N}(\mu_{ik})^m (x_k - v_i)^T (x_k - v_i) / \sum_{k-1}^{N}(\mu_{ik})^m \quad (8)$$

C. Task Buffer-Queues (TBQ)

TABLE II. DETAILS ON CATEGORIES OF BUFFER QUEUES

Buffer-Queues	Task Types	Description
LCMI	SSS + LSS	Light Compute-Memory Intensive
CI	SLS + LLS	Compute Intensive
MI	SSL + LSL	Memory Intensive
HCMI	SLL + LLL	Heavy Compute-Memory Intensive

Table II outlines the categories of buffer-queues following the merging of clusters. The LCMI queue combines SSS and LSS task types, representing tasks that are light in both compute and memory intensity. The CI queue comprises SLS and LLS tasks, focusing on compute-intensive tasks. The MI queue merges SSL and LSL task types, which are more memory-intensive. Lastly, the HCMI queue includes SLL and LLL tasks, characterized as heavy in both compute and memory requirements.

D. Criteria for Performance Assessment

To assess the effectiveness of the clustering algorithms, two key performance metrics are employed: the Classification Rate and the Xie-Beni (XB) Index. These criteria are applied across various fuzzy algorithms to determine their efficiency and accuracy.

Classification Rate: This criterion measures the accuracy with which data points are assigned to their respective clusters. It is calculated by dividing the number of correctly classified data points by the total sample size. The calculation, as defined in Equation (9), involves two components:

1. The numerator represents the intra-cluster compactness, calculated as the mean square distance between each data point and its corresponding cluster center.

2. The denominator is the separation measurement, which is the minimum squared distance between the centroids of different clusters.

$$S_{XB} = \frac{\sum_{i=1}^{c}\sum_{j=1}^{n}\mu_{ij}^{2}\|V_i - X_j\|^2}{\;} \qquad (9)$$

IV. ANALYSIS OF EXPERIMENTAL OUTCOMES

The performance of these clustering techniques was assessed using two primary metrics: the classification rate and the XB index. The experiment was set up with a cluster count of C=8, while the fuzziness index, denoted as 'm', was varied. At an 'm' value of 1.5, both KFCM and GK-FCM outperformed FCM. However, upon altering 'm' to 2 for the same cluster size, GK-FCM exhibited superior performance, achieving a classification rate of approximately 92.87%, surpassing both FCM and KFCM.

TABLE III. COMPARATIVE STUDY OF VARIOUS CLUSTERING TECHNIQUES

Metrics	FCM	KFCM	GK-FCM
Classification Rate (m=2, C=8)	72.5	81.3	92.87
Classification Rate (m=1.5, C=8)	71.3	81.4	88.52
Xie-Beni (XB) Index	0.3212	0.2147	0.0816

Table III provides a comparative analysis of different clustering methods for a cluster count of C=8. It compares the Fuzzy C-means (FCM), Kernel-based Fuzzy C-means (KFCM), and Gustafson-Kessel Fuzzy C-Means (GK-FCM) based on two key metrics: the Classification Rate and the Xie-Beni (XB) Index. The table shows the classification rates for two different values of the fuzziness index 'm' and the XB index, highlighting the superior performance of GK-FCM in terms of higher classification accuracy and a lower XB Index, which indicates better cluster compactness and separation.

The compactness and separation of the fuzzy clusters were also assessed using the Xie-Beni (XB) index. A lower XB value indicates greater separation between clusters, which is desirable. The primary objective of the Fuzzy C-means (FCM) algorithm is to minimize the objective function μ, achievable through reducing the XB value. In the experiments with a fuzziness index of m=2, the XB index (SXB) values for FCM and Kernel-based Fuzzy C-means (KFCM) were 0.3212 and 0.2147, respectively. However, the Gustafson-Kessel Fuzzy C-Means (GK-FCM) demonstrated superior performance with a notably lower SXB index of 0.0816, outperforming the other algorithms in terms of cluster compactness and separation.

The final clustering results, illustrated in Figure 5, reveal that a majority (approximately 82%) of the tasks in the dataset are short-duration tasks requiring small CPU and memory resources. Only about 9% of tasks have high CPU and memory requirements. Around 7% of tasks predominantly utilize large memory, as defined in Table 1, and a smaller portion (about 2%) of the tasks demand high CPU resources.

Figure 5. Clustering results after applying GK-FCM

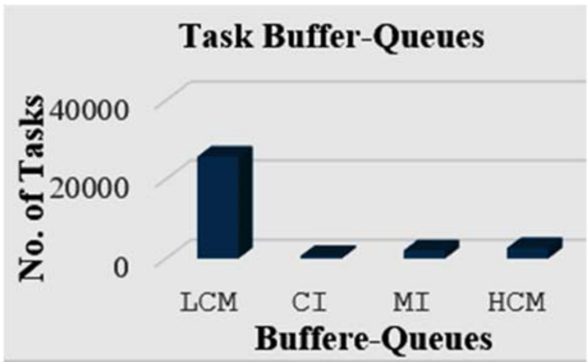

Figure 6. Results of task buffer-queues after grouping the clusters.

During the execution of tasks in cloud environments, it is typical for a task's resource consumption to vary over time. On average, a task T might consume resources at a rate denoted by T_Avg^r over a certain period. However, there are instances during the execution when the resource usage might peak to a higher level, referred to as T_Max^r, exceeding the average rate T_Avg^r.

This method of resource estimation is vital for maintaining efficient and reliable cloud services, especially in scenarios where resource demands can fluctuate significantly.

In this model, T_k^r represents the resource required for task T_k. The standard deviation of the maximum average resource requirement, denoted as T_(Max_Avg)^r, is symbolized by σ, where r refers to resources like CPU and memory. The factor of 2 multiplied with the standard deviation σ is utilized to estimate the resource usage for 95% of all tasks in the clusters, ensuring that the majority of task demands are adequately met.

Taking CPU as one of the resources, Resource Estimation (RE) is computed for all eight clusters. For visualization purposes, sample tasks RE for clusters SSS, SLS, LSS, and LLL are demonstrated in Figure 7. Figure 8 presents a summary of the estimated average CPU usage and the percentage of resource savings for each of the clusters, C1 to C8. According to the estimations, short-duration light tasks consume a maximum of 20% over the planned CPU, resulting in 80% resource savings.

To validate the proposed method's effectiveness in resource savings, a comparison is made with other clustering algorithms such as K-Means (KM), Fuzzy C-Means (FCM), and Kernel-based FCM (KFCM). The comparison, illustrated in Figure 9, showcases the percentage of CPU savings for each individual cluster. The results confirm that the proposed method outperforms existing methods in terms of resource savings, highlighting its efficiency in cloud resource management.

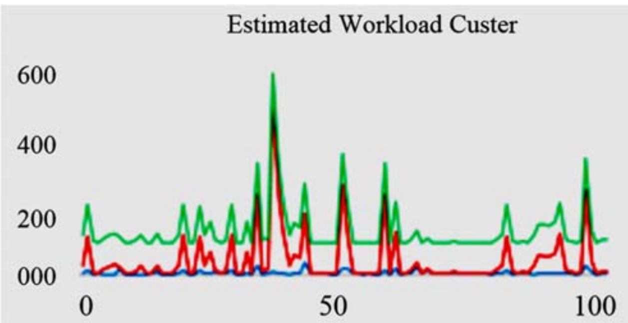

Figure 7. CPU usage of tasks in each cluster with resource estimation.

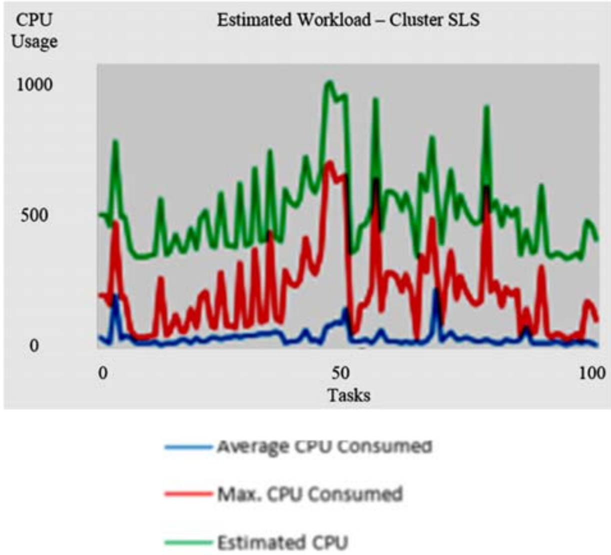

Figure 8. Estimated average CPU usage and the percentage of resource savings for each of the clusters, C1 to C8.

VI. CONCLUSIONS AND FUTURE WORK

The primary aim of this research was to maximize resource utilization, specifically CPU and memory, while ensuring Quality of Service (QoS) in a cloud computing environment. This goal was pursued through the development and implementation of a task classification method utilizing a fuzzy clustering algorithm. The effectiveness of this approach was rigorously tested using real data from the Alibaba dataset, focusing on the accuracy of the clustering model and the CPU resource savings for each cluster.

The experimental results demonstrated a high level of accuracy in the classification rate, achieving an impressive overall resource savings of 85%. Additionally, when compared to existing clustering algorithms, the proposed method showed superior performance in terms of resource savings. This underscores the effectiveness of the proposed approach in optimizing cloud resource management.

A significant outcome of this work is the creation of the Task Buffer-Queues (TBQ), which effectively groups the clustered tasks into categories: Light, Heavy, Compute Intensive, and Memory Intensive. This categorization significantly enhances task scheduling and allocation for Virtual Machines (VMs). Looking ahead, the results obtained from this study will be instrumental in guiding the planning and creation of VMs for task allocation, with a focus on incorporating optimization techniques to further refine and enhance the cloud computing environment.

REFERENCES

[1] Alibaba "Alibaba trace," https:// github.com/ alibaba/ clusterdata, 2017.

[2] Aladwani, Tahani. "Impact of selecting virtual machine with least load on tasks scheduling algorithms in cloud computing." Proceedings of the 2nd international Conference on Big Data, Cloud and Applications. 2017.

[3] Alismaili, et al. (2020). Organizational-level assessment of cloud computing adoption: Evidence from the Australian SMEs. Journal of Global Information Management, 28(2), 73–89. doi:10.4018/JGIM.2020040104

[4] Anupama K C, Shivakumar B R, Nagaraja R, "Task Classification for Improving Scheduling and Resource Management in Cloud Computing" International Journal of Applied Engineering Research ISSN 0973-4562 Volume 17, Number 1 (2022) pp. 41-50

[5] Anupama, K. C., R. Nagaraja, and M. Jaiganesh. "A Perspective view of Resource-based Capacity planning in Cloud computing." In 2019 1st International Conference on Advances in Information Technology (ICAIT), pp. 358-363. IEEE, 2019.

[6] Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum, New York,1981.

[7] Calzarossa, Maria Carla, Luisa Massari, and Daniele Tessera. "Workload characterization: A survey revisited." ACM Computing Surveys (CSUR) 48.3 (2016): 1-43.

[8] Chen, Wenyan, et al. "How does the workload look like in production cloud? analysis and clustering of workloads on alibaba cluster trace." 2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS). IEEE, 2018.

[9] Choi, HeeSeok, et al. "Task classification based energy-aware consolidation in clouds." Scientific programming 2016 (2016).

[10] Di, Sheng, Derrick Kondo, and Franck Cappello. "Characterizing and modeling cloud applications/jobs on a Google data center." The Journal of Supercomputing 69.1 (2014): 139-160.

[11] Elrotub, Mousa, and Abdelouahed Gherbi. "Virtual machine classification-based approach to enhanced workload balancing for cloud computing applications." Procedia computer science 130 (2018): 683-688.

[12] Fahim, Youssef, et al. "Load balancing in cloud computing using meta-heuristic algorithm." Journal of Information Processing Systems 14.3 (2018): 569-589.

[13] Garrison et al. (2012). Success Factors for Deploying Cloud Computing. Commun. ACM. 55, 62–68.

[14] Gustafson, D. E. & Kessel, W. C. "Fuzzy clustering with a fuzzy covariance matrix". In Proc. of IEEE Conf. on Decision and Control including the 17th Symposium on Adaptive Processes, San Diego. pp. 761-766,1979.

[15] Kaur, Ripandeep, and Gurjot Kaur. "Proactive Scheduling in Cloud Computing." Bulletin of Electrical Engineering and Informatics 6.2 (2017): 174-180.

[16] Li, Jian, et al. "Improved FIFO scheduling algorithm based on fuzzy clustering in cloud computing." Information 8.1 (2017): 25.

[17] Li, Lei, et al. "Two-stage adaptive classification cloud workload prediction based on neural networks." International Journal of Grid and High Performance Computing (IJGHPC) 11.2 (2019): 1-23.

[18] Marahatta, Avinab, et al. "Classification-based and energy-efficient dynamic task scheduling scheme for virtualized cloud data center." IEEE Transactions on Cloud Computing (2019).

[19] Mathivanan, Norsyela Muhammad Noor, Nor Azura Md Ghani, and Roziah Mohd Janor. "Improving classification accuracy using clustering technique." Bulletin of Electrical Engineering and Informatics 7.3 (2018): 465-470.

[20] Mishra, Asit K., et al. "Towards characterizing cloud backend workloads: insights from Google compute clusters." ACM SIGMETRICS Performance Evaluation Review 37.4 (2010): 34-41.

[21] Patel, Jemishkumar, et al. "Workload estimation for improving resource management decisions in the cloud." 2015 IEEE Twelfth International Symposium on Autonomous Decentralized Systems. IEEE, 2015. (1

[22] Rimal et al. (2011). Architectural requirements for cloud computing systems: An enterprise cloud approach. Journal of Grid Computing, 9(1), 3–26. doi:10.1007/s10723-010-9171-y

[23] Shahidinejad, Ali, Mostafa Ghobaei-Arani, and Mohammad Masdari. "Resource provisioning using workload clustering in cloud computing environment: a hybrid approach." Cluster Computing 24.1 (2021): 319-342.

[24] Shekhawat, Virendra Singh, Avinash Gautam, and Ashish Thakrar. "Datacenter workload classification and characterization: An empirical approach." In 2018 IEEE 13th International Conference on Industrial and Information Systems (ICIIS), pp. 1-7. IEEE, 2018.

[25] Su, et al. (2009). Shared Services Transformation: Conceptualization and Valuation from the Perspective of Real Options. Decis. Sci. 40, 381–402.

[26] Suresh, Annamalai, and R. Varatharajan. "Competent resource provisioning and distribution techniques for cloud computing environment." Cluster Computing 22.5 (2019): 11039-11046.

[27] Venters et al. (2012). A Critical Review of Cloud Computing: Researching Desires and Realities. J. Inf. Technol. 27, 179–197 (2012).

[28] Xie, Xuanli Lisa, and Gerardo Beni. "A validity measure for fuzzy clustering." IEEE Transactions on pattern analysis and machine intelligence 13.8 (1991): 841-847.

[29] Zhang, PeiYun, and MengChu Zhou. "Dynamic cloud task scheduling based on a two-stage strategy." IEEE Transactions on Automation Science and Engineering 15.2 (2017): 772-783.

[30] Zuo, Liyun, et al. "A multiqueue interlacing peak scheduling method based on tasks' classification in cloud computing." IEEE Systems Journal 12.2 (2016): 1518-1530.

.