International Journal of Innovation and Scientific Research
ISSN 2351-8014 Vol. 73 No. 2 Jul. 2024, pp. 154-163
© 2024 Innovative Space of Scientific Research Journals
http://www.ijisr.issr-journals.org/

K-mean Clustering Scheduling Algorithm for Cloud Computing Containers

Amany AbdElSamea Saeed

Computers and Systems Department, Electronics Research Institute, Cairo, Egypt

ABSTRACT: Containers are effective lightweight virtualization technology due to their common host operating system, rapid launch times, scalability, portability, and quick deployment. With the help of containers, programs can create run-time environment that is independent on the platform that boosts mobility and efficiency by encapsulating all required dependencies (code, running time, system libraries, and tools) into a virtual environment. The scheduler is an essential and critical part of optimizing performance and decreasing the cost of container services. Conventional Scheduling methods which are single-criteria-based algorithms (i.e. First Comes First Serves (FCFS), Round Robin, and Shortest Job First) cannot meet the demands of cloud computing. Multi-objectives-based container scheduling algorithms are therefore required to satisfy efficient resource usage optimization by optimally mapping containers to VMS. A novel methodology to the aforementioned issue may be provided by clustering-based task scheduling algorithms, which facilitate the efficient scheduling of a huge number of containers depending on container multi-criteria. This paper suggests a k-mean clustering algorithm for containers to enhance the load balancing that shortens resource execution times while also boosting the resource utilization rate. According to the experimental findings, the suggested algorithm outperforms FCFS algorithm in with respect to the execution time and maintains significant improvement of resource utilization among virtual machines and physical machines.

KEYWORDS: virtualization, load balancing, containerization, virtual machines, container placement, K-mean Clustering.

1 INTRODUCTION

The Quick development of Computational power, big data, and artificially intelligent (AI) algorithms allows the AI applications to be developed to make people's lives easier. Machine learning (ML) [1] is a subset of AI that automatically allows the machine to autonomously gain knowledge from data, enhance performance from prior experiences and come up with predictions. Data is used to train machine learning algorithms, which then build a model to perform specific task based on this training. These ML algorithms are useful in solving numerous business problems, including clustering, association, regression, and classification [2]. Supervised, semi-supervised, and unsupervised are the three primary categories of machine learning. In Supervised machine learning, machines are trained with "labeled" dataset, and according to the training, the output is predicted using the tested dataset. Classification and regression are examples of supervised learning techniques. Semi-supervised learning [3] combines the benefits of supervised and unsupervised learning. The machine is supplied with a lot of unlabeled input and a limited amount of labeled data. The unlabeled data is utilized to refine the model that the computer learned using the labeled data. Unsupervised learning is a kind of machine learning when no labeled data provided to the computer. The computer is responsible for identifying patterns in the data on its own. In this instance, the data is not labeled, but the algorithm aids the model in creating groups of related data types. Clustering is an example of unsupervised learning techniques.

Clustering involves breaking up the unlabeled data into distinct clusters. The primary objective of the clustering process is to identify and classify groups that share similar characteristics and assign them into clusters. Clustering has a huge number of applications spread across diverse disciplines. Social network analysis, market segmentation, search result grouping, picture segmentation, medical imaging, and cloud computing load balancing are a few of the most widely used applications of clustering. Depending on their cluster model, clustering algorithms fall into one of three categories:

- Connectivity-based clustering (Hierarchical clustering) [3]: The fundamental idea of hierarchical clustering is that items are more closely associated to one another when they are adjacent than when they are farther away. These algorithms use distance-based connections to group "objects" into "clusters". A cluster is identified by the maximum distance required for the connection of the elements of the cluster. The term "hierarchical clustering" is derived from the fact that distinct clusters will develop at varying distances, and these clusters can be visualized using a dendrogram
- Distribution based clustering [4]: It assumes that the data is composed of distributions, such as Normal or Gaussian distributions. The possibility of a point to be part of the distribution is minimized with the increase of the distance from the distribution's center. These models frequently have over-fitting issues
- Density-based Clustering [5]: A density-based clustering technique is essential for identifying nonlinear shapes and structures based on density. Density-Based Clustering defines clusters as regions in the data set which have a higher density than the rest. Density reachability and density connectedness are the guiding principles of density-based algorithms. The most frequently used algorithm in density-based clustering is Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [5]. This algorithm's primary characteristic is its ability to handle noisy input in a single scan and find clusters of any shape
- Graph-based clustering [6]: Graph-based clustering is a straightforward and scalable clustering technique that resembles spectral clustering. Two categories of graph clustering exist the first is "between-graph" where clustering techniques separate a collection of graphs into various clusters and the second technique is "within-graph" where the graph's nodes are grouped together using clustering techniques. There are numerous within-graph clustering algorithms. Power Iteration clustering, Kernel K-means algorithm, are examples of graph-based clustering algorithms
- Center-based clustering [7]: In numerous research domains, center-based clustering methods are extensively employed in unsupervised learning algorithms. A cluster is a collection of items arranged so that each object in the cluster is closer to its "center" than it is to the centers of any other clusters. Samples are grouped using center-based clustering according to a distance measure from the cluster centers. Within this context, a cluster's center may be either a centroid or a medoid. A "centroid" is the average of all the points within a cluster, whereas a medoid is the most representative point of a cluster. A crucial aspect of cluster analysis is selecting an appropriate measure of similarity or dissimilarity (distance), as the effectiveness of clustering algorithms is heavily dependent on this decision. K-mean clustering is the most popular and well-known center-based clustering algorithm. The number of cluster parameters required at the end of these models must be specified in advance; hence prior knowledge of the dataset is crucial. To locate the local optima, these models operate iteratively

Table 1. Comparison between clustering models

Feature	**Hierarchical clustering**	**Distribution based clustering**	**Density-based clustering**	**Center-based clustering**
Concept	Clustering on basis of distance	Clustering on basis of distributions (Normal or Gaussian)	Clustering on basis of densities	Clustering on basis of similarities
Strength	Visual representation of cluster hierarchy, easy to interpret because it can find clusters of any size or shape	Manages complex data structures, varies cluster dimensions and forms, and computes cluster number automatically	Its complexity is fairly low and it identifies outliers effectively	Computationally efficient, Scales to large datasets
Weakness	Computationally expensive, Lack Scalability for handling big datasets, sensitive to data order and distance metric	Computationally expensive and suffer from over-fitting	To find cluster boundaries, they anticipate a density drop of some form	Requires specifying number of clusters, Suffer from clustering data of varying sizes and shapes , affected by noise and outliers
Algorithms example	Agglomerative clustering	Gaussian mixture models	DBSCAN	K-mean clustering

Table 1 presents the comparison between clustering models. An effective clustering algorithm should be able to find clusters with a variety of sizes, densities, and forms. It ought to be capable of handling data noise as well. It shouldn't be affected by the user-inputted parameters or the sequence in which the data is entered. Hierarchical-based clustering is based on the distance, it detects clusters of any shape and size and it provides a visual representation of cluster hierarchy but it is Computationally expensive, lack Scalability for handling big datasets, sensitive to data order and distance metric. Distribution based clustering is based on distributions. It manages complex data structures as well as varying cluster shapes and sizes. but it is computationally expensive and it prone to over fitting. Density-based clustering depends on densities. Its complexity is fairly lower than hierarchical-based clustering and distribution-based clustering and it identifies outliers effectively but a density drop is necessary to identify cluster borders. Finally, center-based clustering is based on similarities. It is the most widely used model since it is computationally efficient and, scales to large datasets but its drawback is that requires specifying number of clusters and it is affected by noise and outliers. Since K-Means clustering technique is a high-performance algorithm, it is the most extensively used and well-known method for clustering data points. Data is expanding exponentially in a cloud environment. Thus, it is critical to manage this enormously heterogeneous data, which may include both organized and unstructured information. In the cloud, this data can be managed on various levels. For cloud computing, it seems that the k-mean clustering job scheduling method is a better option when it comes to virtual machines and containers.

Cloud computing [8] is a distributed computing environment that offers customers online and on-demand networking, storage, and processing as a service. These services are available on a pay-as-you-go basis, with charges made to users based on how much of the resources they utilize through a promising technology called virtualization [9]. Virtualization offers the cloud computing environment with processing power in the form of virtual machines (VM). In a cloud computing environment, resources can scale vertically or horizontally. Vertical scaling involves adding more powerful systems or upgrading to more powerful components. Horizontal scaling involves increasing the number of servers and other processing units. In Virtualization, software constructs an abstraction layer over the hardware where the physical components of a single computer can be separated into several virtual computers by using a tiny layer of software known as a hypervisor. Hypervisor [10] allows multiple operating systems to run simultaneously and share the same physical computer resources. It permits the physical computer (also known as bare metal) to separate its operating system and applications from the host hardware. Hypervisors isolate VMs from one another and allocate processors, memory, and storage between them.

Containers offer a more lightweight and flexible approach to virtualization. Containers are operating system virtualization technique that utilizes the capabilities of the host operating system to separate processes and limit the programmers' CPU, disk space, and memory access. Containerization [11] bundles together all the components required to operate a single application or micro-service (together with the runtime libraries they require) as opposed to setting up a full virtual machine. The operating system as well as all of the code and its dependencies is contained in the container. This makes it possible for apps to operate virtually wherever, including on desktop computers, traditional IT infrastructure, and cloud servers. Workload distribution across containers and virtual machines (VMs) must be done efficiently in the dynamic cloud computing environment, where virtualization and resource sharing are now critical for assuring responsive and seamless service delivery.

A significant and vital part of container orchestration is provided by the scheduler portion [12]. Thus, in cloud data centers, container scheduling and effective run-time management of cloud computing resources become crucial. Conventional scheduling methods which are single-criteria-based algorithms (i.e. First Comes First Serves (FCFS), Round Robin, and Shortest Job First) cannot meet the demands of cloud computing. For efficient resource consumption optimization by mapping containers to VMS appropriately; multi-objectives-based container scheduling algorithms are therefore needed. There are two primary factors for the container management. Initially, a variety of cloud services and apps may be available, each with varying resource needs, such as CPU, disk space, and RAM. It has to do with meeting the needs for several resources. Second, there is a wide range of resource availability across the cloud servers. Improper management of the containers could lead to an imbalance in the load on the cloud servers, making it impossible to provide users with adequate cloud services.

Load balancing enhances the overall performance of the system by equal distribution of the workload among nodes. It guarantees increased resource efficiency and improved user fulfillment by assigning containers to suitable virtual machines (VMs) also by assigning VMs to appropriate host and distributing resource utilization among all hosts. By making the most use of the resources at hand and reducing resource consumption, load balancing allows for scalability, performance maximization, and response time reduction. It has been found that cloud computing and data mining strategies both assist businesses in maximizing profits and reducing expenses in several methods. Since cloud computing works with big data centers, enormous amounts of data must be accessed at once. As a result, it affects the performance. Thus, it appears that using a quick and accurate clustering algorithm is a preferable choice for cloud computing. Clustering is an approach used in to categorize related objects into clusters. Large datasets are clustered quickly and accurately using the k-means clustering algorithm. It entails dividing the given data set into distinct numerical groups known as clusters, with each cluster being linked to a center point (centroid) while assigning every point to the cluster that includes the nearest centroid. K-means clustering aims to partition a group of objects to K clusters so the sum of the squared distances among the objects and the assigned cluster mean is reduced. The k-mean clustering approach for containers is proposed in this paper to improve load balancing and reduce resource execution times while increasing the resource utilization rate for containers and VMs. The proposed approach is compared to the FCFS in terms of execution time and maintains significant improvement of the resource utilization among virtual machines and physical machines. More precisely, the major contribution of this paper might be summed up as follows:

a) Provides a detailed analysis of different types of clustering algorithms and compare between different cluster models
b) Introduces and implements a K-mean clustering algorithm for containers
c) Comparing the result of the proposed algorithm with FCFS algorithm

The format of this paper is as follows: Section II introduces the related work. Section III identifies the basic k-mean clustering algorithm. Section IV presents the proposed k-mean clustering algorithm for containers. Section V covers the implementation and simulation results. Finally, future work and the conclusion are discussed in Section VI.

2 RELATED WORK

This section reviews the most current studies that use the k-mean clustering algorithm in cloud computing environments to allocate resources. Geetha Muthusamy et al. [7] proposes a cluster-based task scheduling (CBTS) framework using K-Means clustering that takes task length and virtual machine capacity into account. In this case, the VMs are categorized according to their processing capacity, and the tasks are grouped according to their length. Following clustering, each cluster's particular work is scheduled to the proper virtual machine inside the VM groups. The goal of the suggested system's dynamic load balancing is to reduce the execution time and makespan. The paper drawback is that the task length is the only factor that forms the clusters. A task's completion time may be restricted by its deadline. As a result, while creating task clusters, the deadline may also be added as a feature of the tasks. This work is done for virtual machines allocation only.

Law S. Xue et al. [13], [14] create a model for batch processing load balancing in cloud computing environments using clustering and Principal Component Analysis (PCA). The utilization of multivariate statistical approaches has been examined in this research, which has helped to build a load balancing strategy. A novel method of taking into account heterogeneous variables that describe physical hosts has therefore been developed, and it is based on PCA. K-means clustering has been used to further determine the variations in the computational resources of the physical host. The clusters were performed using PCA and a clustering algorithm according to the physical host-related parameters such as bandwidth, Random Access Memory (RAM), storage, and Million Instructions per Second (MIPS). Furthermore, the approach has been shown to improve load balancing performance in terms of response time, throughput, makespan, waiting time compared to the Round Robin (RR) and FCFS algorithms. The paper shortage is that it does not use containers to enhance the resource utilization in cloud computing datacenters.

S. Asha et al. [15] utilizes the K-means clustering technique. Based on RAM capacity, the virtual machines are grouped using the k-means clustering technique. In order to maximize processing time, this paper focuses on job scheduling using virtual machine clustering utilizing particle swarm optimization (PSO). An open source cloud platform (CloudSim) was used to model the outcome of cluster-based PSO. Ultimately, the findings showed that the suggested approach has lowered the makespan when contrasted with the current scheduling techniques. The drawback of this paper is that the clustering is based on single criteria only (RAM).

Thilagavathi et al. [16] adopts an agent-based strategy to handle the problem of energy efficiency in a cloud data center through server consolidation, energy-efficient scheduling, and initial VM allocation. The k-means clustering technique is utilized to accomplish the initial virtual machine allocation. The new clustering agent is essential for cutting down on reaction time and energy usage. The frequency of VM migrations, the energy used, and the response time for assigning VMs to incoming requests are used to gauge performance. A clustering agent is used to cluster the hosting servers. It has been noted that effective procedures for choosing target servers and virtual machines aid in achieving load balancing. The suggested approach lowers the number of migrations because of the positive initial VM allocation. Additionally, it demonstrates improvements in response time as the volume of customer queries rises. The disadvantage is that authors didn't address dynamic threshold adjustment and live migration heuristics to enhance the energy efficiency in a cloud computing environment.

Vrajesh Sharma et al. [17] developed a new credits-based scheduling algorithm to improve cloud computing speed and efficiency while removing the shortcomings of the widely used or current method. The suggested system employs the modified K-means clustering technique to classify the cloudlets and virtual machines (VMs), taking into account four real-time parameters: Task-Priority, Task-Length, Cost, and Deadline as credits. The suggested approach has been tested and implemented using CloudSim 3.0.3, a cloud simulation tool. The suggested scheduling algorithm outperformed the priority-based scheduling strategy that was previously in use, as demonstrated by the experimental and simulated findings presented in this study. The paper drawback is that their scheduling technique is not applied on a real cloud container datacenter

In this study, a novel approach using a k-mean clustering algorithm for containers may be offered by task scheduling algorithms based on Multi-criteria based clustering, which enable the effective scheduling of a large number of containers based on container multi-criteria MIPS, RAM, bandwidth, storage. In order to improve load balancing and reduce resource execution times while simultaneously increasing resource utilization rates, this study proposes a k-mean clustering approach for containers.

3 Basic K-Mean Clustering Algorithm

K-means is an unsupervised machine learning technique for data clustering. It can group unlabeled data based on similarities (k) into a predetermined number of clusters. In K-means, the letter "K" denotes the number of clusters the algorithm was able to detect from the available data. The dataset is divided into K pre-defined unique, non-overlapping subgroups, or clusters, via the iterative K-means method, ensuring that each data point belongs to a single group. The objective is to preserve as much space between the clusters and as much similarity as is practical between the intra-cluster data points. The procedure entails clustering data points to minimize the total squared distance between each cluster's centroid, which is the arithmetic mean of all the data points in that cluster. When variation inside a cluster decreases, the homogeneity of data points within the cluster rises. Following are the steps of the basic k-mean clustering algorithm:

1- Indicate the number of clusters, K

2- Select K elements at random from the entire available data sample. These K samples will be used as temporary centroids, commonly known as the mean

3- There are currently (n — k) data samples remaining. Compute the sum of the squared distance between data points and all centroids. If there is a minimal and equal distance between the two clusters then we may select any cluster and insert that sample into it

4- Updating centroids so the new centroids for each cluster will be determined once overall data points are assigned to one. This time, however, the centroid computation will be predetermined. The data points mean allocated to that cluster is used to compute the new centroid. We will have K centroids at this stage, which differ from the randomly selected K samples that were previously chosen

5- Continue iterating until the centroids remain unchanged. i.e., the distribution of data points among clusters remains constant. There will be no or negligible movement of samples among clusters. K-means clusters the data samples into the required number of clusters. The pseudocode of the basic k-mean clustering technique is presented in Algorithm 1

Algorithm 1 Basic K-means Clustering Algorithm

```
Input: Number of clusters (k), n data samples.
Output: Centroid of every discovered cluster, and the assignment of each receiving datum to a cluster.
/K means initialization/
Data-samples = [x1, x2, x3, ..., xn]
Initialize-k-means = [x1, x2, ..., xk]
/Assigning data points within any of the K clusters/
for all (n-k) sample do
    track-minimum-distance
    for all K selected means do
        Compute sample distances from each of the chosen K means.
        Assign the sample in the cluster that has the shortest distance from it.
    end for
end for
  / Updating centroids/
    for all K-means do
        Compute the updated mean values.
    end for
    Repeat the aforementioned loops until the centroid does not update.
K-means clusters the data samples into the required number of clusters.
```

K-Means clustering algorithm is high performance clustering algorithms and so it is the most popular and widely used algorithm for grouping data points into clusters. In a cloud environment, data is growing at an exponential rate. Consequently, it's imperative to manage this massively diverse data, which might be both structured and unstructured. This data can be managed at different levels in cloud. Thus, it appears that using k-mean clustering task scheduling algorithm for containers and virtual machines is a preferable choice for cloud computing. The next section will present the proposed K-mean clustering scheduling algorithm for cloud computing containers

3.1 K-Mean Distance Calculation

Distance measures are crucial for gauging how similar the dataset is to one another. Finding the relationships between data sets, the ways in which different data are similar or unlike from one another and the metrics that are used for comparison are all important. The distance metrics function—which is used to cluster data—is computed for this purpose. This paper uses Manhattan distance and Euclidean distance as follows

- Manhattan distance

The Manhattan distance calculates the absolute differences among two objects' coordinates as shown in eq. 3.1

$$ManhDist = |\, p1 - q1| + |p2 - q2| \qquad \text{(Eq. 3.1)}$$

The generalized n-dimensional space formula is shown in eq 3.2:

$$ManhDist = \sum_{i=1}^{n} |pi - qi| \qquad \text{(Eq. 3.2)}$$

Where,

- n = number of dimensions
- pi, qi = data points

- Euclidean distance

Euclidean Distance denotes the shortest path between two vectors. It is the square root of the total of the squares of the differences between comparable elements as shown in eq. 3.3

$$\boldsymbol{EuclDist} = \sqrt{(\boldsymbol{p1} - \boldsymbol{q1})^2 + (\boldsymbol{p2} - \boldsymbol{q2})^2} \qquad \text{(Eq. 3.3)}$$

For n-dimensional space the formula is shown in eq. 3.4

$$\boldsymbol{EuclDist} = \sqrt{\sum_{i=1}^{n} (\boldsymbol{pi} - \boldsymbol{qi})^2} \qquad \text{(Eq. 3.4)}$$

4 K-Mean Clustering Scheduling Algorithm For Cloud Computing Containers

For effective resource usage optimization, multi-objectives-based container scheduling algorithms must map containers to VMS in an optimal manner. Clustering is a technique used for enhancing cloud computing resource utilization. One crucial step in raising cloud computing's overall performance is task scheduling. To cut down on processing time, clustering-based task scheduling methods, which enable the effective scheduling of a large number of containers based on container multi-criteria, may offer a promising approach to the aforementioned problem. The containers and virtual machine clustering method makes use of the K-means clustering algorithm. Based on MIPS, RAM, and BW capacity, the containers and virtual machines are grouped using the k-means clustering technique. In order to improve load balancing and reduce resource execution times while simultaneously increasing resource utilization rates, this study proposes a k-mean clustering approach for containers

The pseudocode of the proposed K-mean clustering scheduling algorithm for cloud computing containers is presented in Algorithm 2.

Algorithm 2 K-mean Clustering Scheduling Algorithm for Cloud Computing Containers

```
Input: Container list, VM list, Number of clusters (k), n data samples.
Output: Allocation of containers on VMs
/Cloudsim initialization/
Initialize CloudSim by creating the Datacenter Broker, containers, virtual machines and cloudlets.
Provide the unallocated container list and unassigned VM list to the Datacenter broker

/Clustering the containers using k means /
for each container in containerlist
        Get the values of MIPS and RAM and bandwidth size of each container
end for
Apply Modified k-means clustering algorithm on the containers and aggregate them into three clusters high, medium, and low

/ Clustering the VMs using k means
for each VM in VMlist
        Get the values of MIPS and RAM and bandwidth size of each VM
end for
Apply Modified k-means clustering algorithm on the VMs and aggregate them into three clusters high, medium, and low
for each container in containerlist
Assign container to VM of appropriate cluster
vindex++
if Vindex=VmListSize
{
Vindex =0;
}
end for
Apply descending sort on container and VMs in clusters.
Mapping of the container to the VM
for each container do
   Assign container to VM of appropriate cluster
         Increment the VM
         if VMmax >= ListSize
            {
                VNindex=0;
            }
end for
sendNow (container id, virtual machine id)
```

Firstly, the Datacenter Broker, virtual machines, and cloudlets are created as part of the initialization of the CloudSim simulator. Datacenter Broker receives the list of unallocated containers and unassigned VMs. The Modified K-means clustering algorithm is utilized to categorize the containers and further dividing them into three categories. Low, medium, and high, in decreasing order. Datacenter broker calculates each virtual machine’s processing capacity based on RAM, size, bandwidth, and MIPS and deploys the K-means clustering technique on VMs to group or classify them into three categories low, medium and high then assign the containers to VM of appropriate cluster.

5 Implementation And Simulation Results

This section verifies the suggested placement strategies, which are subsequently assessed using ContainerCloudSim simulator.

5.1 Implementation Environment

ContainerCloudSim [12] offers support for simulation and modeling of cloud computing infrastructures with containers including interfaces for managing the CPU, memory, and storage resources of hosts, virtual machines (VMs), containers, and data centers. Features that let researchers compare and experiment with different container scheduling and provisioning strategies. Studying how well provisioning algorithms allocate resources in an energy-efficient manner. Support for simulation scalability is necessary since there are far more containers in Container-as-a-Service (CaaS) environment than there are virtual machines in a data center.

5.2 Performance Metrics

Several experiments were conducted and analyzed using the following metrics in order to provide a thorough examination of the recommended algorithm to compare and assess the performance of the algorithm:

Simulation Time (ST): It is the amount of time, measured in seconds, that is spent performing an experiment. It is the overall time needed to finish a set of tasks.

Throughput: is the number of jobs that a computer system will complete in a predetermined period of time. The overall performance of a system is assessed based on its throughput.

CPU Utilization: It is a relative indicator of the amount of non-idle processing occurring. The amount of real-time processing is shown by the ratio of the amount of time spent on non-idle processing.

5.3 Parameter Setting

We assess the effectiveness of our suggested K-mean clustering scheduling technique for cloud computing containers using the ContainerCloudSim platform and compare it to alternative task scheduling algorithms. The setting of the parameters of the proposed algorithm which provide us the best performance is illustrated in Table 2. We set the number of clusters of the containers and VMs to 3 categories low, medium and high then the containers are assigned to the appropriate VM cluster. We use Euclidean distance and the Manhattan distance for distance calculation

Table 2. ContainerCloudSim Parameter Setting

Type	Parameters	Value
Containers	*CONTAINER_TYPES CONTAINER_MIPS* *CONTAINER_PES* *CONTAINER_RAM* *CONTAINER_BW*	3 4658, 9320, 18636 1 128, 256, 512 2500
Virtual Machine	*VM_TYPES* *VM_PES* *VM_RAM* *VM_BW* *VM_SIZE*	4 2, 4, 1, 8 1024, 2048, 4096, 8192 100000 2500
Hosts	*HOST_TYPES* *HOST_MIPS* *HOST_PES* *HOST_RAM* *HOST_BW* *HOST_STORAGE*	3 37274 4, 8, 16 65536, 131072, 262144 1000000 1000000

5.4 Experimental Results

A comparison is made between our k-mean clustering scheduling technique for cloud computing containers and the First-Come-First-Served (FCFS) approach [17]. Requests are automatically processed by FCFS and are queued in the order that they are received. It is the simplest CPU scheduling algorithm that is currently available.

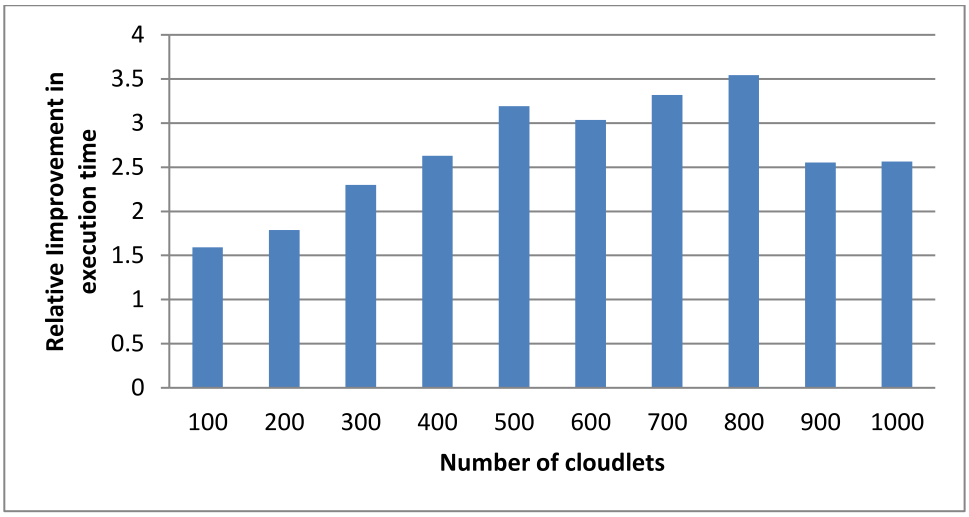

Fig. 1. Relative improvement in execution time of the proposed k-mean algorithm w.r.t FCFS algorithm

The relative increase in execution time of the suggested k-mean approach for containers in comparison to the FCFS algorithm is displayed in Fig. 1. It is demonstrated that the suggested approach minimizes the needed time while achieving a good balance of system loads. We see that when the number of cloudlets grows, the relative response time rises linearly. It performs best when there are 800 cloudlets because the execution time advantage over the FCFS method is just 3.5%. The relative improvement in execution time is 3.2%, 3%, and 3.4% for 500, 600, and 700 cloudlets, respectively. When the number of the cloudlets is 100 the relative improvement in execution time is the worst since it is about 1.6%.

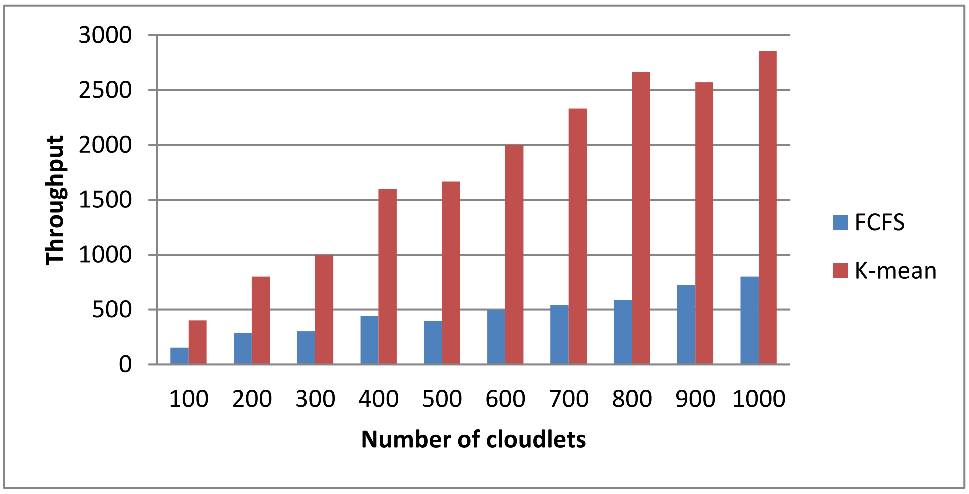

Fig. 2. Throughput

Fig. 2 illustrates that the suggested technique outperforms FCFS in terms of throughput across all cloudlets. A portion of the cloudlets that we sent were successful, while some were unsuccessful. The suggested technique performs best when there are 1000 cloudlets, which allows for the optimal placement of containers on virtual machines (VMs) with lower response times and higher throughput. When there are only 100 cloudlets, the algorithm performs worst.

6 CONCLUSION

For effective resource use optimisation, multi-objectives-based container scheduling algorithms must map containers to VMS in an optimal manner. Clustering-based task scheduling methods, which enable the effective scheduling of a large number of containers based on container multi-criteria, may offer a fresh approach to the aforementioned problem. This paper suggests a k-mean clustering technique for containers in order to enhance load balancing, decrease resource execution times, and increase resource utilization rates at the same time. The experimental findings demonstrate that, in terms of execution time and throughput, the suggested approach outperforms the FCFS algorithm. According to the experimental findings, the suggested method outperforms alternative algorithms by a margin of 35%. Instead of using simulation in the future, we can implement the suggested k-mean allocation technique on an actual platform. In order to optimize the placement of containers on virtual machines, we can also experiment with various machine learning methods.

REFERENCES

[1] N.Valarmathy and S.Krishnaveni, «Performance Evaluation and Comparison of Clustering Algorithms used in Educational Data Mining», International Journal of Recent Technology and Engineering (IJRTE), Vol. 7, pp. 103–113, April 2019.

[2] C. Zhang, W. Huang, and T. Niu, «Review of Clustering Technology and Its Application in Coordinating Vehicle Sub systems», Automot. Innov., vol. 6, pp. 89–115, 2023. https://doi.org/10.1007/s42154-022-00205-0.

[3] K.Kameshwaran, K.Malarvizhi, « Survey on Clustering Techniques in Data Mining», International Journal of Computer Science and Information Technologies (IJCSIT), Vol. 5, no. 2, pp. 2272-2276, 2014.

[4] P. H. Ahmad and S. Dang, «Performance Evaluation of Clustering Algorithm Using Different Datasets», Journal of Information Engineering and Applications, Vol.5, No.1, pp. 36–49, 2015.

[5] A. Fahim, «A varied density-based clustering algorithm», Journal of Computational Science, Vol. 66, 2023, https://doi.org/10.1016/j.jocs.2022.101925.

[6] A. Sarmiento, I. Fondón, Iván Durán-Díaz, and Sergio Cruces, «Centroid-Based Clustering with αβ-Divergences», Entropy, vol. 21, no. 2, 2019. https://doi.org/10.3390/e21020196.

[7] G. Muthusamy, S. R. Chandran, «Cluster-based Task Scheduling Using K-Means Clustering for Load Balancing in Cloud Datacenters», Journal of Internet Technology, Vol. 22, No.1, pp. 121–130, 2021.

[8] A. Priyadarshini, S. K. Pradhan, S. Pattnaik, S. R. Laha, and B. K. Pattanayak, «Dynamic Task Migration for Enhanced Load Balancing in Cloud Computing using K-means Clustering and Ant Colony Optimization», International Journal on Recent and Innovation Trends in Computing and Communication, vol. 11, no. 7, pp. 156–162, July, 2023.

[9] A. Abdelsamea, S. M. Nassar, and H. Eldeeb, «The past, present and future of scalable computing technologies trends and paradigms: A survey,» International Journal of Innovation and Applied Studies, vol. 30, no. 1, pp. 199–214, July 2020.

[10] V. Sharma, M. Bala, «A Credits Based Scheduling Algorithm with Kmeans Clustering», First International Conference on Secure Cyber Computing and Communication (ICSCCC), pp. 82–86, 2018.

[11] A. M. Hafez, A. Abdelsamea, A. A. El-Moursy, S. M. Nassar and M. B. E. Fayek, «Modified Ant Colony Placement Algorithm for Containers,» 2020 15th International Conference on Computer Engineering and Systems (ICCES), Cairo, Egypt, pp. 1-6,2020, doi: 10.1109/ICCES51560.2020.9334671.

[12] W. Zhang, L. Chen, J. Luo, and J. Liu, «A two-stage container management in the cloud for optimizing the load balancing and migration cost», Future Generation Computer Systems, vol. 135, pp. 303–314, https://doi.org/10.1016/j.future.2022.05.002

[13] L. S. Xue, N. Abd Majid, E. A. Sundararajan, «A Principal Component Analysis and Clustering based Load Balancing Strategy for Cloud Computing», TEM Journal, vol. 9, no. 1, pp. 93-100, 2020, https://doi.org/10.18421/TEM91-14.

[14] L. S. Xue, N. A. Abd Majid, and E. A. Sundararajan, «Dynamic Virtual Machine Allocation Policy for Load Balancing using Principal Component Analysis and Clustering Technique in Cloud Computing», Journal of Telecommunication, Electronic and Computer Engineering, Vol. 10, No. 3-2, pp. 47–52, 2018.

[15] S.Asha, P.Deepa, R.Gowtham, K. Kousalya, « Cluster based Task Scheduling in Cloud Environment using PSO Framework», International Journal of Advanced Research Trends in Engineering and Technology (IJARTET), Vol. 4, pp. 12–18, March, 2017.

[16] T. Narayanamoorthy, D. D. Dharani, R. Sasilekha, V. Suruliandi, and V. R. Uthariaraj, «Energy efficient load balancing in cloud data center using clustering technique,» International Journal of Intelligent Information Technologies, vol. 15, no. 1, pp. 84–100, 2019.

[17] V. Sharma, M. Bala, «An Improved Task Allocation Strategy in Cloud using K-means Clustering Technique», Egyptian Informatics Journal, Vol. 21, pp. 201-208, 2020.https://doi.org/10.1016/j.eij.2020.02.001

[18] S. F. Piraghaj, A. V. Dastjerdi, R. N. Calheiros, and R. Buyya, « ContainerCloudSim: An environment for modeling and simulation of containers in cloud data centers», SOFTWARE: PRACTICE AND EXPERIENCE, vol. 47, pp. 505–521, 2017.