The Scientific Temper (2024) Vol. 15 (spl): 107-113

E-ISSN: 2231-6396, ISSN: 0976-8653

Doi: 10.58414/SCIENTIFICTEMPER.2024.15.spl.13

https://scientifictemper.com/

RESEARCH ARTICLE

Distribution of virtual machines with SVM-FFDM approach in cloud computing

D. Jayadurga*, A. Chandrabose

Abstract

Virtual machine (VM) distribution in cloud computing plays a pivotal role in optimizing resource allocation and improving overall system performance. This study proposes a novel approach for efficient VM distribution using a combination of support vector machine (SVM) and the finest fit decreasing modifier (FFDM) algorithm. SVM is employed to classify and predict resource utilization patterns, ensuring that VMs are allocated based on predicted workloads. The FFDM algorithm, a modified version of the traditional first fit decreasing (FFD) algorithm, is then applied to optimize the packing of VMs onto physical servers by minimizing resource wastage and enhancing load balancing. By integrating machine learning techniques with optimization algorithms, the proposed approach achieves a more effective VM allocation strategy, leading to improved system efficiency, reduced energy consumption, and enhanced scalability in cloud environments. Simulation results demonstrate the superior performance of the SVM-FFDM method compared to traditional VM allocation techniques in terms of resource utilization and operational cost.

Keywords: Cloud computing, Virtual machine, Support vector machine, Finest fit decreasing modifier.

Introduction

The advent of cloud computing has revolutionized the way organizations deploy, manage, and utilize their computing resources. Among the critical components of cloud architecture is the concept of virtualization, which allows multiple virtual machines (VMs) to operate on a single physical server. This abstraction layer enables efficient resource allocation, scalability, and flexibility, making it a cornerstone of modern IT infrastructure. The distribution of virtual machines across physical servers is a pivotal aspect that significantly influences the performance, reliability, and cost-effectiveness of cloud services, Shi, F., & Lin, J. (2022), Supreeth, S., Patil, K., Patil, S. D., Rohith, S., Vishwanath, Y., & Prasad, K. V. (2022), Ullah, A., & Nawi, N. M. (2023), Ullah, A., Nawi, N. M., & Ouhame, S. (2022).

Virtual machines are software-defined instances that emulate physical computers equipped with their own operating systems and applications. The distribution of VMs is essential for optimizing resource utilization and ensuring that computational tasks are executed efficiently. By strategically allocating VMs across a cloud environment, providers can balance workloads, enhance system performance, and minimize energy consumption. This process involves the consideration of several factors, including resource availability, application requirements, user demands, and geographical locations of data centers, Han, B., & Zhang, R. (2022), Belgacem, A. (2022).

One of the primary motivations for the distribution of VMs is the need for load balancing. In a cloud environment, workloads can vary dramatically based on user activity, time of day, or specific events. When a particular server becomes overloaded, distributing VMs across less-utilized servers can help maintain optimal performance levels and prevent bottlenecks. Load balancing algorithms play a crucial role in this process, employing various strategies such as round-robin, least connections, and resource-based methods to allocate VMs efficiently.

Furthermore, the distribution of VMs is closely tied to fault tolerance and disaster recovery strategies. In a

Edayathangudy G.S Pillay Arts and Science College (Autonomous) (Affiliated to Bharathidasan University, Tiruchirappalli), Nagapattinam, Tamil Nadu, India.

***Corresponding Author:** D. Jayadurga, Edayathangudy G.S Pillay Arts and Science College (Autonomous) (Affiliated to Bharathidasan University, Tiruchirappalli), Nagapattinam, Tamil Nadu, India, E-Mail: jayaddd23@gmail.com

How to cite this article: Jayadurga, D., Chandrabose, A. (2024). Distribution of virtual machines with SVM-FFDM approach in cloud computing. The Scientific Temper, **15**(spl):107-113.

Doi: 10.58414/SCIENTIFICTEMPER.2024.15.spl.13

Source of support: Nil

Conflict of interest: None.

Received: 10/09/2024 **Accepted:** 22/09/2024 **Published:** 16/10/2024

cloud setting, the failure of a single physical machine can lead to significant service disruptions. By distributing VMs across multiple servers, organizations can ensure that if one server fails, the workloads can be redirected to other functioning machines, thereby maintaining service continuity. This redundancy is essential for meeting service level agreements (SLAs) and ensuring high availability for users, Mandal, R., Mondal, M. K., Banerjee, S., Srivastava, G., Alnumay, W., Ghosh, U., & Biswas, U. (2023).

Another key consideration in VM distribution is geographical distribution, which is critical for minimizing latency and enhancing user experience. By deploying VMs in data centers located closer to end-users, cloud providers can significantly reduce the time it takes for data to travel between servers and clients. This geographical distribution not only improves response times but also allows organizations to comply with data residency requirements in various jurisdictions, Saidi, K., & Bardou, D. (2023); Khan, M. S. A., & Santhosh, R. (2022).

Moreover, the rise of multi-cloud and hybrid-cloud strategies has further complicated the distribution of VMs. Organizations increasingly leverage multiple cloud providers to enhance redundancy and optimize costs. In such environments, effective VM distribution strategies are necessary to manage resources across diverse platforms, ensuring seamless operation and interoperability, Supreeth, S., Patil, K., Patil, S. D., Rohith, S., Vishwanath, Y., & Prasad, K. V. (2022).

Virtual Machine Allocation In Cloud Computing

Cloud computing has transformed the way businesses and individuals access and utilize computing resources. At the heart of this paradigm shift lies the concept of virtualization, which enables the creation of multiple virtual machines (VMs) on a single physical server. This technology allows for better resource management, increased flexibility, and improved scalability. However, the effectiveness of cloud computing heavily relies on the efficient allocation of these virtual machines, Madhusudhan, H. S., Gupta, P., Saini, D. K., & Tan, Z. (2023).

Virtualization emerged in the 1960s but gained significant traction in the 1990s with the development of more sophisticated hypervisors, which allowed multiple operating systems to run on a single hardware platform. The introduction of technologies such as VMware and Microsoft Hyper-V laid the foundation for modern cloud computing. By the 2000s, major cloud service providers, including Amazon web services (AWS), Google cloud platform (GCP), and Microsoft Azure, began leveraging virtualization technologies to offer scalable, on-demand computing resources.

The allocation of virtual machines is a critical aspect of cloud management. Effective VM allocation enhances resource utilization, reduces operational costs, and improves performance. In a cloud environment, resources such as CPU, memory, storage, and network bandwidth must be efficiently distributed among various VMs to meet application demands. The allocation process involves placing VMs on physical servers based on factors like resource requirements, service-level agreements (SLAs), and workload characteristics, Singh, G., Rani, L., Ghosh, P., Goyal, S., & Vajpayee, A. (2022, December).

Finest Fit Decreasing Modifier (FFDM) Algorithm

As cloud computing continues to evolve, efficient allocation of VMs is paramount for optimizing resource utilization and ensuring high performance. The finest fit decreasing modifier (FFDM) algorithm is an enhancement of the traditional finest fit decreasing (FFD) algorithm, designed to address the challenges associated with VM allocation in cloud environments. This algorithm aims to minimize resource wastage while ensuring that VMs are allocated in a manner that meets the varying demands of applications, Sundas, A., Badotra, S., Alotaibi, Y., Alghamdi, S., & Khalaf, O. I. (2022).

In cloud computing, resource allocation involves distributing available physical resources (CPU, memory, storage) among multiple VMs to ensure optimal performance. Traditional approaches, such as the first fit and best fit algorithms, have limitations in terms of resource utilization and fragmentation. The FFD algorithm, which sorts VMs based on their resource requirements before allocating them to the smallest available physical server that meets their needs, improves upon these limitations by reducing fragmentation. The FFDM algorithm builds upon this foundation, incorporating additional modifiers to further enhance the allocation process, Ullah, A., & Chakir, A. (2022).

The FFDM algorithm can be broken down into the following steps:

Step 1: Input parameters

The algorithm begins by collecting input parameters, including the list of VMs to be allocated, their resource requirements (CPU, memory, etc.), and the list of available physical servers with their resource capacities.

Step 2: Sorting

The VMs are sorted in decreasing order based on their resource requirements. This step ensures that larger VMs are allocated first, which is a critical factor in reducing fragmentation and improving overall utilization.

Step 3: Finest fit allocation

The algorithm iteratively selects each VM from the sorted list and attempts to allocate it to the most appropriate physical server:

- It searches for the server with the smallest available capacity that can still accommodate the VM's

requirements. This is the "finest fit" aspect, which minimizes wasted resources on the selected server.

- If multiple servers can accommodate the VM, the algorithm may apply additional criteria, such as load balancing, to select the server that will ensure optimal performance across the cloud environment.

Step 4: Modifier implementation

The "modifier" aspect of FFDM introduces specific criteria to adjust the allocation process. This could involve:

Load balancing

Modifying the allocation to prevent overloading any single server, ensuring that workloads are distributed evenly across the available infrastructure.

- *Energy efficiency*

Incorporating energy consumption metrics into the decision-making process so that servers with lower energy costs are prioritized when allocating VMs.

- *Predictive analytics*

Utilizing historical data to predict future resource needs and adjusting allocations accordingly.

Step 5: Iteration

The algorithm continues this process for all VMs in the sorted list until all are allocated or no suitable servers remain.

Step 6: Output

The final output is the mapping of VMs to physical servers, along with any relevant performance metrics (e.g., resource utilization, energy consumption).

Support Vector Machine

Support vector machine (SVM) is a powerful supervised machine learning algorithm primarily used for classification and regression tasks. Developed in the 1990s, SVM has gained popularity due to its effectiveness in high-dimensional spaces and its ability to model complex relationships between features. SVM aims to find the optimal hyperplane that separates different classes in the feature space, making it particularly useful for tasks involving binary classification, Dhahi, S. H., Dhahi, E. H., Khadhim, B. J., & Ahmed, S. T. (2023); Arunkumar, M., & Kumar, K. A. (2023); Arunkumar, M., & Kumar, K. A. (2023).

Step 1: Hyperplane

In n-dimensional space, a hyperplane is a flat affine subspace of dimension n-1. For example, in a 2D space, the hyperplane is a line, while in a 3D space, it is a plane. The hyperplane serves as the decision boundary that separates different classes.

Step 2: Support vectors

Support vectors are the data points that lie closest to the hyperplane and are critical for defining its position. They are the points that influence the orientation and position of the hyperplane. SVM uses only these support vectors to construct the decision boundary, making it robust to outliers.

Step 3: Margin

The margin is defined as the distance between the hyperplane and the nearest support vectors from either class. SVM aims to maximize this margin, as a larger margin typically indicates better generalization to unseen data.

Step 4: Kernel Trick

SVM can efficiently perform classification in high-dimensional spaces through the use of kernel functions. The kernel trick allows SVM to implicitly map input data into higher-dimensional feature spaces without having to compute the coordinates of the data in that space. Common kernel functions include:

- Linear Kernel: $\boldsymbol{K}(x, y) = x.y$
- Polynomial Kernel: $K(x, y) = (x.y + C)^d$
- Radial Basis Function (RBF) Kernel: $K(x, y) = e^{-\gamma\|x-y\|^2}$.

Step 5: Soft margin

In practice, data is often not perfectly separable. The soft margin SVM introduces a penalty for misclassifications, allowing some data points to be on the wrong side of the margin. This flexibility helps in handling noise and ensures better generalization.

Proposed SVM-FFDM Approach

Efficient VM allocation is critical for optimizing resource utilization and performance in cloud computing environments. The proposed approach combines the finest fit decreasing modifier (FFDM) algorithm with support vector machine (SVM) to enhance the allocation process. By integrating these two techniques, we aim to achieve a robust VM allocation strategy that minimizes resource fragmentation while effectively predicting workload requirements based on historical data.

This proposed approach outlines a step-by-step procedure that combines the FFDM algorithm with SVM for efficient VM allocation in cloud computing environments.

Step 1: Data Collection

Historical data

Collect historical data regarding VM usage, including:

- Resource requirements (CPU, memory, storage)
- Workload characteristics (type of application, request patterns)
- Performance metrics (response time, resource utilization)

Step 2: Feature Selection for SVM

Identify features

Analyze the historical data to identify relevant features that impact VM allocation, such as:

- Time of day (peak and off-peak hours)
- Types of applications running (resource-intensive vs. lightweight)
- Historical resource consumption patterns

Step 3: SVM Model Development

Data preprocessing

Normalize and preprocess the selected features to ensure consistency and suitability for SVM training.

Train the SVM

Split the data into training and testing sets. Train the SVM model using the training data to predict future resource requirements based on the identified features.

Model Validation

Validate the SVM model using the testing set to ensure its accuracy in predicting resource demands. Adjust parameters as necessary to improve performance.

Step 4: Predict Resource Requirements

Input incoming workloads

For each new VM request, use the trained SVM model to predict resource requirements based on its characteristics.

Classification

Classify each incoming workload into categories (e.g., high, medium, low resource demand) based on the predicted requirements.

Step 5: Implement the FFDM Algorithm

Initialize parameters

Prepare the list of available physical servers along with their current resource capacities.

Sort VMs

Sort the predicted VMs in decreasing order based on their predicted resource requirements obtained from the SVM model.

Step 6: Finest Fit Allocation Process

Iterate through VMs

For each VM in the sorted list:

Find suitable server

Identify the physical server with the smallest available capacity that can accommodate the VM's resource requirements.

Allocation criteria

If multiple servers meet the requirements:

- Apply load balancing criteria to prevent server overload.
- Consider energy efficiency metrics to prioritize servers with lower energy costs.

Allocate VM

Assign the VM to the selected physical server and update the server's resource capacity accordingly.

Step 7: Modifier Integration

Load balancing

Ensure that the allocation promotes even distribution of workloads across all available servers.

Energy efficiency

Implement energy-efficient practices by considering the power consumption of servers during the allocation process.

Step 8: Monitor and Adapt

Performance monitoring

Continuously monitor the performance of allocated VMs and servers, tracking metrics such as resource utilization and response times.

Retrain SVM model

Periodically retrain the SVM model with new historical data to improve accuracy and adapt to changing workload patterns.

Result and Discussion

The performance of the proposed SVM-FFMD approach is evaluated with the existing VM allocation methods like first fit (FF), best fit (BF), and first fit decreasing (FFD) using different evaluation metrics like resource utilization, average response time, energy consumption and latency for the different number of hosts and various number of VMs.

Simulation Setup

Hosts

- Number of Hosts: 20
- CPU Cores per Host: 16
- RAM per Host: 128 GB
- Storage Capacity per Host: 2 TB
- Network Bandwidth: 10 Gbps
- Power Consumption per Host: 400W

Virtual machines

- Number of VMs: 300
- CPU Cores per VM: 2
- RAM per VM: 8 GB
- Storage per VM: 50 GB
- Network Bandwidth per VM: 500 Mbps
- VM Lifespan: 8 hours
- Workload Type: Mixed (CPU-bound, memory-bound, I/O-bound)

Table 1: Resource utilization (in %) obtained by the proposed SVM-FFDM approach with FF, BF and FFD with number of hosts = 20 and Number of VMs = 200

Task size	*Resource utilization (in %)*			
	Proposed SVM-FFDM	*FF*	*BF*	*FFD*
100	85.5	78	82.5	80
200	88	76.5	81	79.5
300	90.2	75	79	78
400	92.5	73.5	77	76
500	93.8	71	75.5	74.5
600	94.5	69.5	74	72
700	95	68	72.5	70
800	95.5	66.5	71	68.5

Table 2: Average response time (in ms) obtained by the proposed SVM-FFDM approach with FF, BF and FFD with number of hosts = 20 and Number of VMs = 200

Task size	*Average response Time (in ms)*			
	Proposed SVM-FFDM	*FF*	*BF*	*FFD*
100	120.5	135	130	128.5
200	115	140	125.5	133
300	110	145.5	128	135.5
400	105.5	150	130.5	140
500	100	155	135	142.5
600	95	160	138	145
700	92.5	165.5	140	148.5
800	90	170	142.5	150

Performance Analysis with Number of Hosts = 20 and Number of VMs = 200

Table 1 depicts the resource utilization (in %) obtained by the proposed SVM-FFMD and existing FF, BF, and FFD with a number of hosts = 20 and a number of VMs = 200. From Table 1, As the task size increases from 100 MB to 800 MB, the proposed SVM-FFDM approach demonstrates consistently higher resource utilization compared to the existing methods (FF, BF, FFD). The results indicate that the proposed approach effectively manages resources, reducing fragmentation and optimizing VM allocation, particularly with larger task sizes.

Table 2 depicts the average response time (in ms) obtained by the proposed SVM-FFDM approach existing FF, BF, and FFD with a number of hosts = 20 and number of VMs = 200. From Table 2, The proposed SVM-FFDM approach consistently demonstrates lower average response times across all task sizes compared to existing methods (FF, BF, FFD). As the task size increases from 100 MB to 800 MB, the response times for all approaches generally increase; however, the SVM-FFDM remains the most efficient. This indicates that the proposed approach optimally allocates resources and manages workloads, resulting in improved responsiveness for VM requests.

Table 3 depicts energy consumption (in kWh) obtained by the proposed SVM-FFDM approach existing FF, BF, and FFD with a number of hosts = 20 and number of VMs = 200. From Table 3, The proposed SVM-FFDM approach consistently demonstrates lower energy consumption across all task sizes compared to existing methods (FF, BF, FFD). As the task size increases from 100 MB to 800 MB, energy consumption increases for all approaches; however, the SVM-FFDM remains the most energy-efficient. This indicates that the proposed approach optimizes resource allocation effectively, resulting in reduced energy usage during the VM allocation process.

Table 4 depicts latency obtained by the proposed SVM-FFDM approach existing FF, BF, and FFD with a number of hosts = 20 and a number of VMs = 200. From Table 4, the proposed SVM-FFDM approach consistently demonstrates lower latency across all task sizes compared to existing methods (FF, BF, FFD). As the task size increases from 100 MB to 800 MB, latency generally increases for all approaches; however, the SVM-FFDM remains the most efficient. This indicates that the proposed approach effectively manages task allocations, leading to reduced latency in processing requests.

Table 3: Energy consumption (in kWh) obtained by the proposed SVM-FFDM approach with FF, BF and FFD with number of hosts = 20 and Number of VMs = 200

Task size	*Energy consumption (in kWh)*			
	Proposed SVM-FFDM	*FF*	*BF*	*FFD*
100	150	165	160	158
200	145	170	158.5	162
300	140	175.5	155	164.5
400	135	180	152	168
500	130	185	150	170
600	125	190	148	173
700	120	195.5	145.5	175.5
800	115	200	143	178

Performance Analysis with Number of Hosts = 20 and Number of VMs = 300

Table 5 depicts the resource utilization (in %) obtained by the proposed SVM-FFMD and existing FF, BF, and FFD with a number of hosts = 20 and a number of VMs = 300. From Table 5, The proposed SVM-FFDM approach consistently demonstrates higher resource utilization compared to the existing methods (FF, BF, FFD) across all task sizes. As the task size increases from 100 to 800 MB, the SVM-FFDM shows a gradual increase in resource utilization, indicating effective resource management and reduced fragmentation. The existing methods show a declining trend in resource utilization efficiency as task sizes increase, suggesting potential limitations in their allocation strategies.

Table 4: Latency (in (ms)) obtained by the proposed SVM-FFDM approach with FF, BF and FFD with number of hosts = 20 and Number of VMs = 200

Task size	*Latency (in (ms))*			
	Proposed SVM-FFDM	*FF*	*BF*	*FFD*
100	50	60	58	57
200	48	62.5	56	59
300	45	65	55	61
400	43	67	54	62
500	40	70	52	64
600	38	73	51	65
700	36	75	50	66
800	34	78	49	68

Table 5: Resource utilization (in %) obtained by the proposed SVM-FFDM approach with FF, BF and FFD with number of hosts = 20 and Number of VMs = 300

Task size	*Resource utilization (in %)*			
	Proposed SVM-FFDM	*FF*	*BF*	*FFD*
100	87.5	80	83	81
200	90	78	82.5	79.5
300	92	75	80	78
400	93.5	73	78	76.5
500	94.5	70	76.5	74
600	95	68	75	72
700	95.5	66	73.5	70.5
800	96	64.5	72	68

Table 6 depicts the average response time (in ms) obtained by the proposed SVM-FFDM approach existing FF, BF, and FFD with a number of hosts = 20 and a number of VMs = 300. From Table 6, The proposed SVM-FFDM approach consistently demonstrates lower average response times across all task sizes compared to existing methods (FF, BF, FFD). As the task size increases from 100 to 800 MB, the response times generally increase for all approaches; however, the SVM-FFDM remains the most efficient. This indicates that the proposed approach effectively manages task allocations, leading to improved responsiveness in processing requests.

Table 7 depicts energy consumption (in kWh) obtained by the proposed SVM-FFDM approach existing FF, BF, and FFD with a number of hosts = 20 and a number of VMs = 300. From Table 7, The proposed SVM-FFDM approach consistently demonstrates lower energy consumption across all task sizes compared to existing methods (FF, BF, FFD). As the task size increases from 100 to 800 MB, energy consumption tends to increase for all approaches; however, the SVM-FFDM remains the most energy-efficient solution. This indicates that the proposed approach optimally allocates resources and manages workloads, leading to reduced energy usage during the VM allocation process.

Table 6: Average response time (in ms) obtained by the proposed SVM-FFDM approach with FF, BF and FFD with number of hosts = 20 and Number of VMs = 300

Task size	*Average response time (in ms)*			
	Proposed SVM-FFDM	*FF*	*BF*	*FFD*
100	110.5	125	120	118.5
200	105	128.5	123	120
300	100	132	125.5	122.5
400	95	135.5	128	125
500	90	140	130.5	127.5
600	85	144.5	133	130
700	82	148	135.5	132.5
800	80	150	138	135

Table 7: Energy consumption (in kWh) obtained by the proposed SVM-FFDM approach with FF, BF and FFD with number of hosts = 20 and Number of VMs = 300

Task size	*Energy consumption (in kWh)*			
	Proposed SVM-FFDM	*FF*	*BF*	*FFD*
100	140	155	150	148
200	135	160	152.5	155
300	130	165	155	158
400	125	170	157.5	160
500	120	175	160	162.5
600	115	180	162.5	165
700	110	185	165	167.5
800	105	190	167.5	170

Table 8: Latency (in (ms)) obtained by the proposed SVM-FFDM approach with FF, BF and FFD with number of hosts = 20 and Number of VMs = 300

Task size	*Latency (in (ms))*			
	Proposed SVM-FFDM	*FF*	*BF*	*FFD*
100	45	55	52	50
200	43	57.5	53	51
300	40	60	54	52
400	38	62.5	56	53.5
500	36	65	57.5	55
600	34	67.5	58.5	56
700	32	70	59	57
800	30	72	60	58.5

Table 8 depicts latency obtained by the proposed SVM-FFDM approach existing FF, BF, and FFD with a number of hosts = 20 and number of VMs = 300. From Table 8, The proposed SVM-FFDM approach consistently demonstrates lower latency across all task sizes compared to existing methods (FF, BF, FFD). As the task size increases from 100 MB

to 800 MB, the latency generally increases for all approaches; however, the SVM-FFDM remains the most efficient in minimizing response times. This indicates that the proposed approach effectively manages task allocations, leading to improved responsiveness in processing requests.

Conclusion

In this study, we proposed the support vector machine with the finest fit decreasing modifier (SVM-FFDM) approach for VM allocation in cloud computing environments. The performance of the proposed method was evaluated against existing strategies, namely FF, BF, and FFD, using various metrics, including resource utilization, average response time, energy consumption, and latency.

Resource Utilization

The proposed approach achieved higher resource utilization rates, effectively minimizing resource wastage. This efficiency is critical in cloud environments where maximizing resource usage is essential for reducing operational costs and enhancing overall system performance.

Average Response Time

The SVM-FFDM method consistently recorded lower average response times compared to the existing algorithms. This improvement indicates better task scheduling and allocation, resulting in faster processing of user requests and improved user experience.

Energy Consumption

The proposed approach exhibited lower energy consumption levels, showcasing its effectiveness in reducing the environmental footprint associated with cloud operations. This aspect is increasingly important as organizations strive for sustainability in their IT infrastructure.

Latency

The SVM-FFDM method achieved reduced latency across varying task sizes, indicating superior handling of task requests. Lower latency contributes to enhanced system responsiveness and can lead to better performance for time-sensitive applications.

In conclusion, the SVM-FFDM approach provides a robust solution for VM allocation, outperforming traditional methods in critical performance areas. The findings underscore the potential of integrating machine learning techniques, such as support vector machines, with optimization algorithms to enhance cloud computing resource management. Future work could explore further refinements of the SVM-FFDM approach and its applicability in diverse cloud scenarios, including hybrid and multi-cloud environments.

References

Arunkumar, M., & Kumar, K. A. (2023). GOSVM: Gannet optimization based support vector machine for malicious attack detection in cloud environment. *International Journal of Information Technology*, 15(3), 1653-1660.

Belgacem, A. (2022). Dynamic resource allocation in cloud computing: analysis and taxonomies. *Computing*, 104(3), 681-710.

Dhahi, S. H., Dhahi, E. H., Khadhim, B. J., & Ahmed, S. T. (2023). Using support vector machine regression to reduce cloud security risks in developing countries. *Indonesian Journal of Electrical Engineering and Computer Science*, 30(2), 1159-1166.

Han, B., & Zhang, R. (2022). Virtual Machine Allocation Strategy Based on Statistical Machine Learning. *Mathematical Problems in Engineering*, 2022(1), 8190296.

Khan, M. S. A., & Santhosh, R. (2022). Hybrid optimization algorithm for vm migration in cloud computing. *Computers and Electrical Engineering*, 102, 108152.

Madhusudhan, H. S., Gupta, P., Saini, D. K., & Tan, Z. (2023). Dynamic virtual machine allocation in cloud computing using elephant herd optimization scheme. *Journal of Circuits, Systems and Computers*, 32(11), 2350188.

Mandal, R., Mondal, M. K., Banerjee, S., Srivastava, G., Alnumay, W., Ghosh, U., & Biswas, U. (2023). MECpVmS: an SLA aware energy-efficient virtual machine selection policy for green cloud computing. *Cluster Computing*, 26(1), 651-665.

Saidi, K., & Bardou, D. (2023). Task scheduling and VM placement to resource allocation in Cloud computing: challenges and opportunities. *Cluster Computing*, 26(5), 3069-3087.

Shi, F., & Lin, J. (2022). Virtual machine resource allocation optimization in cloud computing based on multiobjective genetic algorithm. *Computational Intelligence and Neuroscience*, 2022(1), 7873131.

Singh, G., Rani, L., Ghosh, P., Goyal, S., & Vajpayee, A. (2022, December). Artificial Intelligence Based Virtual Machine Allocation and Migration Policy using Improved MBFD. *In 2022 IEEE International Conference on Current Development in Engineering and Technology (CCET)* (pp. 1-6). IEEE.

Sundas, A., Badotra, S., Alotaibi, Y., Alghamdi, S., & Khalaf, O. I. (2022). Modified Bat Algorithm for Optimal VM's in Cloud Computing. *Computers, Materials & Continua*, 72(2).

Supreeth, S., Patil, K., Patil, S. D., Rohith, S., Vishwanath, Y., & Prasad, K. V. (2022). An Efficient Policy-Based Scheduling and Allocation of Virtual Machines in Cloud Computing Environment. *Journal of Electrical and Computer Engineering*, 2022(1), 5889948.

Supreeth, S., Patil, K., Patil, S. D., Rohith, S., Vishwanath, Y., & Prasad, K. V. (2022). An Efficient Policy-Based Scheduling and Allocation of Virtual Machines in Cloud Computing Environment. *Journal of Electrical and Computer Engineering*, 2022(1), 5889948.

Ullah, A., & Chakir, A. (2022). Improvement for tasks allocation system in VM for cloud datacenter using modified bat algorithm. *Multimedia Tools and Applications*, 81(20), 29443-29457.

Ullah, A., & Nawi, N. M. (2023). An improved in tasks allocation system for virtual machines in cloud computing using HBAC algorithm. *Journal of Ambient Intelligence and Humanized Computing*, 14(4), 3713-3726.

Ullah, A., Nawi, N. M., & Ouhame, S. (2022). Recent advancement in VM task allocation system for cloud computing: review from 2015 to2021. *Artificial Intelligence Review*, 55(3), 2529-2573.