Received: March 11, 2024. Revised: May 31, 2024.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Membrane Computing-based Resource Scheduling Using Meta-Heuristic Firefly Optimization Algorithm for Cloud Environment

Visalaxi G[1]* **Muthukumaravel A[2]**

[1]*Department of Computer Science and Engineering,*
Bharath Institute of higher Education and Research, Chennai, India 600 073.
[2]*Department of Arts and Science, Bharath Institute of Higher Education and Research, Chennai, 600 073, India*
* Corresponding author's Email: vesalaxi53@outlook.com

Abstract: Cloud Computing (CC) offers clients to access services in a variety of situations under the management of a separate cloud service provider. The CPU, network, and storage are each given specific and distinct responsibilities through resource scheduling. Users must wait for resources to become available while tasks are being provided in a cloud environment, which causes longer wait times, and a lower quality of service for cloud users. To overcome these challenges, this paper proposes a novel MEmbrane Computing based RE source Scheduling in Cloud (MERE-CLOUD) approach for effective resource scheduling with the advent of membrane computing and evolutionary techniques to enhance resource allocation. Initially, the user provides the task to the network to check the resource availability in the cloud environment and the received tasks are prioritized by using Membrane Spiking Neural Network (MSNN). The Firefly Optimization Algorithm (FOA) is implemented to allocate the resources in cloud based on dynamic prediction policy. Finally, the allocated task will be assigned to the virtual machine which enhances the Quality of Service (QoS) and improves availability and reliability with lower error rates. The performance of the proposed MERE-CLOUD approach is evaluated based on throughput, fitness function and task execution parameters on different cloud resources and achieves an accuracy of 94.72% which is highly reliable for task prioritization.

Keywords: Cloud computing, Membrane spiking neural network, Firefly optimization algorithm, Resource scheduling, Membrane computing.

1. Introduction

Cloud computing is an effective way to offer on-demand software through the internet. Cloud computing has a number of advantages for instance the quality of medical services can be improved by using sensor cloud and Internet of Things (IoT) technology to provide adequate health care methods [1-3]. With the rise of cloud computing, it's become more important to improve task scheduling approaches and algorithms for massive data workloads. In cloud computing, it provides a prior solution which emphasizes the significance of good massive data job scheduling, which exacerbates data processing [4, 5].

Resource allocation to cloud clients is a complex process since it is difficult to achieve optimal resource allocation, which entails profit maximization and the deft allocation of scarce resources [6, 7]. Users may speed up the deployment of workflow apps and create scaling strategies with dynamic resource allocation. Replicas are required in order to transfer several resources to the cloud infrastructure at once from a shared resource pool [8-10].

Intrusions, malware, and attempts to collect personal data for personal gain are among the security issues that affect networks and machines in public online environments. To safeguard data and sharing, security rules are necessary across several cloud system tiers [11, 12]. Distributed environment, virtualizes huge machines and can form a virtual cluster if there are several virtual machines. In light of variations in computing by various users, this will offer flexibility and environment adaptation [13].

Table 1. Notations

Notation	Description
w_{ij}	Connection weight between i and j
τ	Pulse's width
d_k	Postsynaptic neuron's waking time
k	Presynaptic neuron's active connection
$u_j(t)$	Neuron j's potential output
β	Discount factor
t	Time
n	Film areas
I	Region
d	Dimensions of search space
t	Next task
y	Previous task
F	Firefly
M_t^k	Unassigned nodes
o_{ty}^{α}	Number of fireflies on the membrane object set
α	Constant factor
m'	Layers of film

Due to the global dispersion of services, changing work load situations, and diverse cloud client needs, big data task scheduling in cloud systems is a time-consuming process [14]. The difficulties that degrade the cloud environment to the customer include increased processing time in a virtual network and low QoS for cloud users [15]. There is a chance of both internal and external attacks while managing important data, transactions, and public messages, which could affect overall performance [16, 17]. To overcome these challenges, this research proposes a novel MEmbrane Computing based RE source Scheduling in Cloud (MERE-CLOUD) approach for utilizing the benefits of membrane computing to enable intelligent resource scheduling techniques for cloud computing. The major contribution of the research are as follows,

- Initially, the user provides the task to the network to check the resource availability in the cloud environment. The received tasks are prioritized by using Membrane Spiking Neural Network (MSNN).
- The prioritized tasks are fed to the Firefly Optimization Algorithm (FOA) to allocate the resources in cloud based on dynamic prediction policy. A dynamic prediction rule provides solutions based on analyzing various uncertainties within a given timeframe.
- By assigning particular micro-clouds to particular virtual machines (VMs), it enhances Quality of Service (QoS), improves availability and reliability with lower error rates to fulfil reliability requirements.
- The performance metrics of the proposed MERE-CLOUD approach is evaluated based on throughput, fitness function and task execution parameters on different cloud resource nodes under overloaded PMs, VM selection policy outputs, and assessment of the new resource management model.

The following research is organized as follows: The literature review of methods that are currently in use is covered in Section 2. Section 3 provides a detailed description of the proposed approach and associated membrane calculations. Section 4 offers a thorough explanation of the findings along with a discussion of the proposed approach. Eventually, Section 5 concludes the research with future scope.

2. Literature survey

An integral component of cloud platform resource management is task scheduling. In-depth analyses and research have been presented in numerous titles. These researchers base their optimization objectives and scope on problem features are given below.

In 2021 Chakravarthi, K.K. and Shyamala, L., [18] suggested a TOPSIS inspired budget and deadline aware multi-workflow scheduling for cloud computing. The suggested T-BDMWS method is evaluated using the following metrics such as Cloud-Based Workflow Scheduling Algorithm (CWSA), Budget Heterogeneous Early Finish Time (BDHEFT), Budget Heterogeneous Early Finish Time (BHEFT), and Resource consumption algorithm. However, scheduling with deadline is a more challenging issue for cloud systems.

In 2022 Otair, M., et al [19] suggested Use the multi-sentence ad optimizer to enhance cloud task scheduling. Using the best and second-best solutions available, the IMOMVO technique dynamically improves the AP updating equation to address the mean positioning (AP) problem. When tested with various datasets, the suggested technique produces runtimes of less than 186.33 seconds for 100 tasks and 934.92 seconds for 600 tasks. The throughput is 0.19 and the processing power Vm is 0.25 kW after 100 operations are completed. However, the resource efficiency remains an issue.

In 2022 Abualigah, L., et al [20] suggested PSO swarm intelligence for cloud computing IoT task scheduling applications with Aquila optimizer. The task scheduling issue should be resolved and the benefits should be maintained by using a hybrid CAE transformation method to get the required changes between search operators. By comparing the suggested hybrid CAE approach with parameters such Wilcoxon signed rank test, post hoc analysis, and maximum, average, and minimum predicted completion times, results were obtained. However, the suggested algorithm has slow learning process.

In 2023 Maroosi, A. and Muniyandi, R.C., [21] suggested an innovative multiverse optimization method inspired by membranes for building cloud web services with SLAs that take QoS into account. This approach introduces the Membrane-Inspired Multiverse Optimization (PMIMVO) algorithm, which divides the variables into subgroups for various types of membranes. The suggested PMIMVO approach can increase integrated quality of service (QoS) by up to 38% when compared to current techniques. However, low accuracy is observed for tasks with very small length.

In 2023 Purba Daru Kusuma and Ashri Dinimaharawati [22] suggested Extended Stochastic Coati Optimizer. The ESCO has three references in its guided search such as the global best unit, a randomly selected unit, and a randomized unit within the search space. ESCO is challenged to solve 23 classic functions and benchmarked with five shortcoming metaheuristics such as GPA, POA, GSO, ASBO, and COA in solving 13, 21, 23, 16, and 13 functions, respectively. However, the efficiency of the suggested ESCO approach is very low.

In 2023 P. Kusuma and A. Dinimaharawati [23] suggested a New Optimization Method and Its Hyper Strategy Investigation. FDSA is designed as a directed search-based metaheuristic without deploying any neighbourhood search. These four references are the best member which is the resultant of three shuffled members within the swarm; a shuffled member within the swarm; and the resultant of the best member, a shuffled member within the swarm, and the corresponding member. However, the performance of FDSA, a fully executed search metaheuristic is still mere in some functions.

In 2023 P. D. Kusuma and A. Novianty [24] suggested a Metaheuristic in Which Each Agent Interacts with All Other Agents. TIA is a swarm intelligence which relies on the interaction among solutions in the population. The core and distinct concept of TIA is that each solution interacts with all other solutions in every iteration to find the best possible solution. However, the performance of TIA is need to be improved for solving the fixed dimension multimodal problems.

In 2024 Purba Daru Kusuma and Meta Kallista[25] suggested a Swarm-based Metaheuristic Enriched with Crossover Technique and Unbalanced Neighbourhood Search. In MCA, the global finest solution becomes the reference in the first step while the middle between two stochastically chosen solutions becomes the reference in the second step. The neighbourhood search is performed in the third step. However, developing better neighborhood search and crossover-based search will be challenging.

The majority of recent research has been on distributed computing, including task scheduling issues. By considering a majority of researches, resource utilisation, overall cost for completing all user actions, execution time, power consumption, and fault tolerance are the challenges raised as a result of growing resource consumption in cloud data centres. To combat these issues, a novel MERE-CLOUD approach is proposed for effective resource scheduling in cloud environments using membrane computing techniques.

3. Membrane computing based resource scheduling in cloud

In this research, a novel MEmbrane Computing based REsource Scheduling in Cloud (MERE-CLOUD) method is proposed for enabling intelligent scheduling techniques for cloud computing. The fundamental purpose of task scheduling algorithms is to accomplish three significant objectives, namely resource minimization and task scheduling that permits the construction of physical systems. Initially, the user provides the task to the network to check the resource availability in the cloud. The received tasks are prioritized by using Membrane Spiking Neural Network (MSNN). The prioritized tasks are fed to the Firefly Optimization Algorithm (FOA) to allocate the resources in cloud based on dynamic prediction policy. A dynamic prediction rule provides solutions based on analyzing various uncertainties within a given timeframe. By assigning particular micro-clouds to particular virtual machines (VMs), it enhances Quality of Service (QoS), improves availability and reliability with lower error rates to fulfil reliability requirements. The general block diagram of the proposed MERE-CLOUD method is shown in Figure 1.

3.1 Task prioritization using spiking neural network

A sort of Spiking Neural Network model (SNN) is a way of neurons communication which consist of spikes, or brief electrical impulses, are primarily employed for neuronal transmission. In terms of dynamics, SNN systems are a part of the third generation of neural network methods. In order to combine the concepts of dealing with single objects and storing information in the amount of time that has

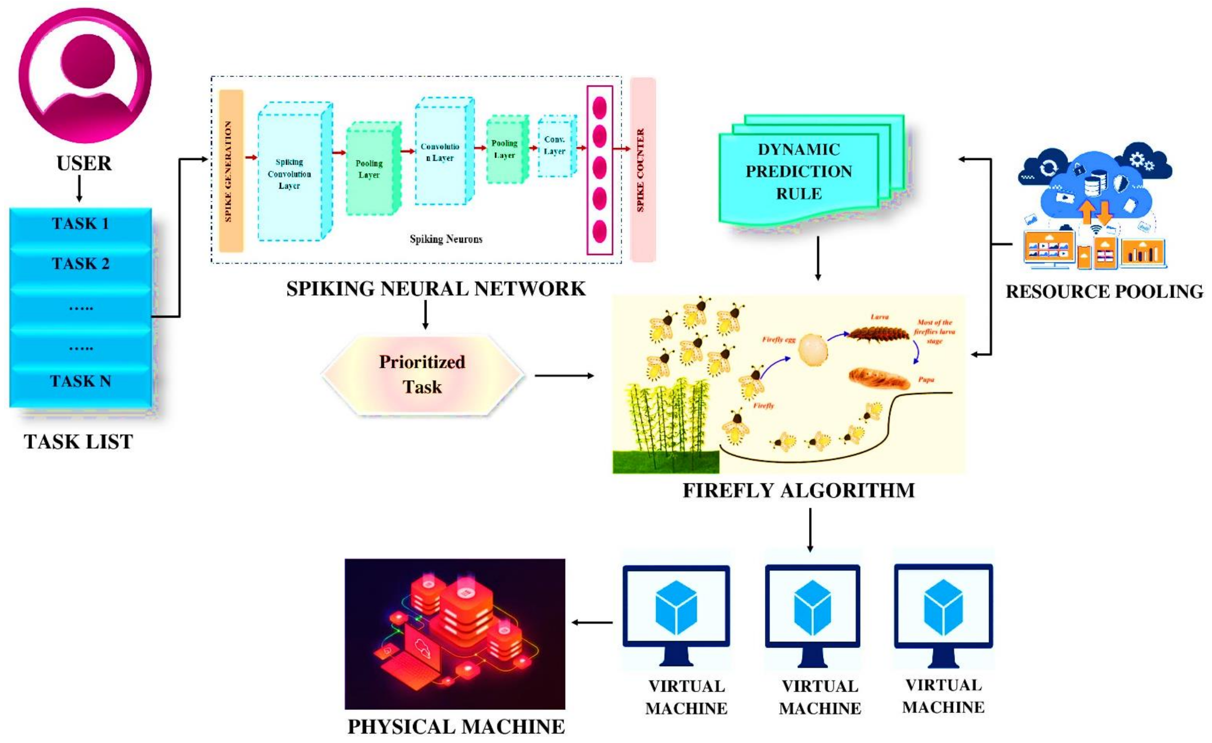

Figure. 1 Block diagram of the proposed methodology

passed between the spikes involved. A group of neurons that incorporate delays compose the components of the SNN architecture. A neuron, j, is a member of a set where w_{ij} is the connection weight between i and j:

$$x_j(t) = \sum_{i\in\Gamma_j} w_{ij}\, \varepsilon(t - t_i) \tag{1}$$

Synaptic potential may be computed using equation (2):

$$\varepsilon(t) = \frac{t}{T} e^{1-\frac{t}{T}} \tag{2}$$

The pulse's width is defined by the constant τ. Equation (3) therefore describes the number of synaptic connections of neuron j:

$$x_j(t) = \sum_{i\in\Gamma_j} \sum_{k=1}^{m} w_{ij}^{k}\, y^{k}(t) \tag{3}$$

The neuron's output is described as follows:

$$y_j^k(t) = \varepsilon(t - t_i - d_k) \tag{4}$$

where d_k is the amount of time that separates the postsynaptic neuron's waking time from the presynaptic neuron's active connection (k). Equation 5 finally expresses the neuron j's output potential $u_j(t)$:

$$u_j(t) = \sum_{t_j^{(f)}\in F_j} \eta\left(t - t_j^{(f)}\right) + \sum_{i\in\Gamma_j} \sum_{t_i^{(g)}} w_{ij}\, \varepsilon(t - t_i^{(g)} - d_{ij} \tag{5}$$

Were,

$$F_j = \{t^{(f)}; 1 \le f \le n\} = \{t | u_j(t) = \vartheta\} \tag{6}$$

There has been n recorded pulses. In the case of SNN, three-dimensional kernel programming is required. Conversely, the perceptron approach makes use of the simplest one- or two-dimensional architecture. Stated differently, all of the neuron outputs of each layer may be estimated concurrently because of its straightforward activation function shape. However, because of its exponential structure, the simultaneous calculation of the time variable using mathematical approximations, and the fact that the neurons' output varies, the SNN activation function is more difficult.

3.2 Dynamic predictor rule

A set of prediction solutions based on various internal or external uncertainty analyses throughout time are provided by dynamic prediction rules. In the dynamic hard case, the objective function is typically implemented as a succession of instantaneous functions. u:X $\times$ C $\rightarrow \mathbb{R}^n$ which is continuous and restricted. The objective function with the discount factor β is given by equation 7.

$$\sum_{t=0}^{T} \beta^t \mu(x_t, c_t) \tag{7}$$

Therefore, $Z_t \in ZCR^m$, according to IAHC, represents the uncertainty condition that will be satisfied in the random variable model. Since the distribution of the random variable at time t+1 only varies in accordance with its value at time t, Pr ($Z_{t+1} \leq Z|Z_t, Z_{t-1} \ldots.) = P_r(Z_{t+1} \leq Z|Z_t)$. The equation $Q(Z', Z) = \Pr(Z_{t+1} \leq Z'|Z_t = Z)$ is used to represent stochastic processes. Furthermore, we presume that someone is aware of Z_t value at time t.

Additionally, a modified version of the standard moving average technique assesses the predicted ultimate consumption of the CPU, network bandwidth, and RAM. In addition, stale approximations refer to outdated mean values for a specific time series window size. In order to calculate the end-user approximation for CPU, RAM, and Network Bandwidth, the estimate of the old value and the estimate of the new value, which is equal to equation 8, 9, and 10, are combined. Maximum dynamic optimization is anticipated.

$$\widehat{U}_{CPU} = k \times \frac{\sum_{i \in window1} U_{cpu}^{i}}{size(window1)} + (1-k) \times \frac{\sum_{i \in window2} U_{cpu}^{i}}{size(window2)} \tag{8}$$

$$\widehat{U}_{RAM} = k \times \frac{\sum_{i \in window1} U_{RAM}^{i}}{size(window1)} + (1-k) \times \frac{\sum_{i \in window2} U_{RAM}^{i}}{size(window2)} \tag{9}$$

$$\widehat{U}_{NET} = k \times \frac{\sum_{i \in window1} U_{NET}^{i}}{size(window1)} + (1-k) \times \frac{\sum_{i \in window2} U_{NET}^{i}}{size(window2)} \tag{10}$$

Where, $\widehat{U}_{CPU}$, $\widehat{U}_{RAM}$ or $\widehat{U}_{NET}$ indicates for well-known moving average algorithms, the factor k serves as a defined constant that predicts the predictive value of CPU, network bandwidth, and RAM. In more detail, Ui CPU, Ui NET, and Ui RAM reflect the used values for calendar, and k specifies the values of the new sample estimate and the old sample estimate for the projected resource utilization of the resource type. CPU, RAM, and network utilization, in that order. bandwidth. Our work includes multiple criteria such as CPU, network bandwidth, and RAM. The utilization definition for high workload risk has now been scaled back and given priority for resource allocation.

3.3 Firefly optimization algorithm

Firefly algorithm is a recently developed swarm intelligence system utilised mostly for the optimization of numerical problems. Each firefly's Fi = (Fti1, Fti2..., Ftid) position is regarded as an object and all fireflies that match each part's Fi = (Fti1, Fti2..., Ftid) are considered as an object and the firefly corresponds with the solution set is viewed as the membrane system object set. Each region's many sets can be stated in equation (11) as follows:

$$\text{Li} = (\text{FiF2i... Fni}), \text{I} = 1, 2, 3, 5, \ldots \tag{11}$$

where n is represented in film areas I and Ft1i, Ft2i, ..., Ftni symbolises the solution for each particle in region I in which Ftni = (ftni1, ftni2...), and d is the dimensions of the space to be searched. Tasks would be assigned with the mapping matrix to a single data centre at a time, where each row represents a single item of one value. The proposed approach collects tasks and data centre ID for the input data. The output resulting from the tasks transition to the designated data centres should be considered to map the data centres to the tasks.

Each firefly chooses classification thresholds and then uses the interclass variance criterion to analyse the final solution. Membrane object-set provides probable answers for the discovery of the next task t and the previous task y, like Firefly F, by using the following equation (12):

$$F_{ty}^{k} = \begin{cases} \frac{o_{ty}^{\alpha}}{\sum l \epsilon M_t^k \tau_{ty}^{\alpha}}, if\ y \epsilon N_t^k \\ 0, \qquad if\ y \notin N_t^k \end{cases} \tag{12}$$

Here,

M_t^k represents unassigned nodes

o_{ty}^{α} represents number of fireflies on the membrane object set and α is a constant factor. The overall flow diagram of the firefly optimization algorithm is depicted in Figure 2.

The firefly iterates to assign tasks based on each iteration which is shown in equation (13).

$$o_{ty} \leftarrow (1-\rho) o_{ty}, \forall (t, y) \in E \tag{13}$$

Structure of a membrane system in which the master film and auxiliary film are defined. At least one firefly appears in each film. The original membrane system's structure is as follows: wherein

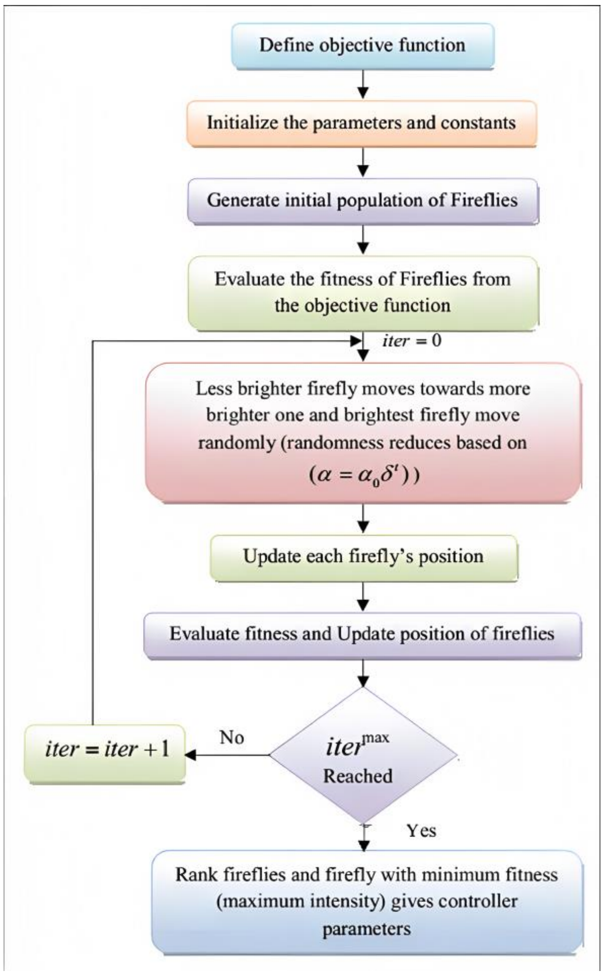
Define objective function
Initialize the parameters and constants
Generate initial population of Fireflies
Evaluate the fitness of Fireflies from the objective function
$iter = 0$
Less brighter firefly moves towards more brighter one and brightest firefly move randomly (randomness reduces based on $(\alpha = \alpha_0 \delta^t)$)
Update each firefly's position
Evaluate fitness and Update position of fireflies
$iter^{max}$ Reached
No
Yes
$iter = iter + 1$
Rank fireflies and firefly with minimum fitness (maximum intensity) gives controller parameters

Figure. 2 Flow diagram of Firefly optimization

the structure has m'= 6 layers of film, including the surface film 1, the membrane 5, and the auxiliary film 3, 4, and 5 The surface film, or film 1, does not perform the particular fitness calculation because it is simply responsible for recycling the membrane object set abandoned by the main film and it is represented in equation (14) as:

$$a0 = \lambda \;;\; a1 = F1F2F3 \ldots Fw1, \; w1<w \;;\; a2 = Fw1 + 1Fw1 + 2 \ldots Fw2, \; w1+w2<w \tag{14}$$

The objective function should be changed to reflect the changing locations of the firefly once they have all moved to the brighter set. The new estimate should be compared to the previous one to see if the new location is superior to the previous one. If a firefly passes by a place that appears to be better than any other detected locations, but it or any other firefly does not end up there by the last iteration of the algorithm for any reason, that best site is still logged. Following the re-evaluation, the algorithm performs an update for the selected firefly before moving on to the next firefly each at a time. In a membrane objective set, where each firefly is given a unique starting point in the related search space, a cluster of fireflies will determine the best solution concurrently, but in a membrane, object set, the solution is merged with rapid task assignment.

4. Results and discussion

The simulations were run using the CloudSim platform using an Intel i5 processor, 16 GB of RAM, and Windows 10 installed. The comparison took throughput, functional suitability, and task performance as the evaluation metrics. The proposed MERE-CLOUD framework and the existing PMIMVO [21], IMOMVO [19], T-BDMWS [18], and Hybrid IAO [20] are suggested approaches. With additional iterations, the algorithm's convergence increased for some of the solved problems, producing more ideal solutions and matching allocation schemes. As a result, resource planning takes less time to complete the overall tasks.

4.1 Performance evaluation

The experimental data are used to evaluate the research are F1 score, recall, accuracy, and precision. The statistical analysis of the parameters is shown below.

$$\text{Accuracy} = \frac{TP+TN}{total\ no.of\ samples} \tag{15}$$

$$recall = \frac{TP}{TP+FN} \tag{16}$$

$$f1\ score = 2\left(\frac{precision*recall}{precision+recall}\right) \tag{17}$$

$$Sensistivity = \frac{TP}{TP+FP} \tag{18}$$

$$TRP = \frac{TP}{TP+FP} \tag{19}$$

The number of true negatives, false negatives, false positives and true positives are indicated by the letters TN, FN, FP, TP and FP respectively. The performance of the forecast is improved by raising the accuracy value.

Figure 3 shows the accuracy curve with accuracy and loss on both vectors. The MERE-CLOUD method's accuracy increases with increasing accuracy. The accuracy versus loss curve in Figure 4 demonstrates how, as accuracy are improved, the model's loss decreases. So, the predicted accuracy of 94.72% for the proposed MERE-CLOUD is highly reliable for task prioritization.

Received: March 11, 2024. Revised: May 31, 2024.

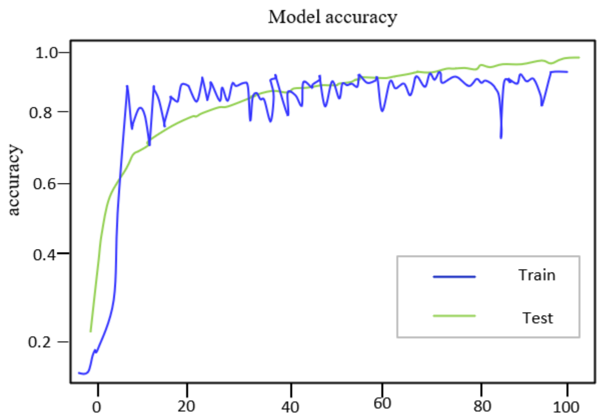

Figure. 3 Performance curve of the proposed MERE-CLOUD Model

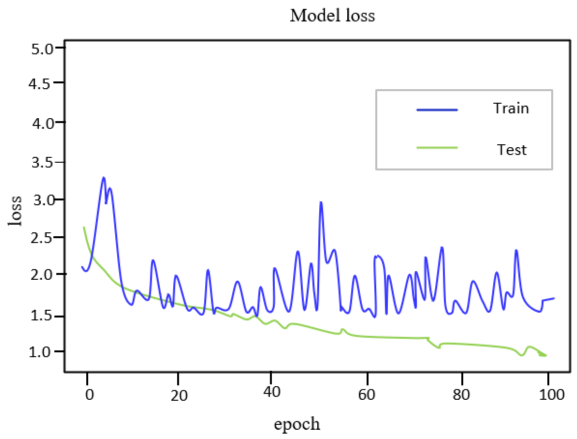

Figure. 4 Loss curve of the proposed MERE-CLOUD model

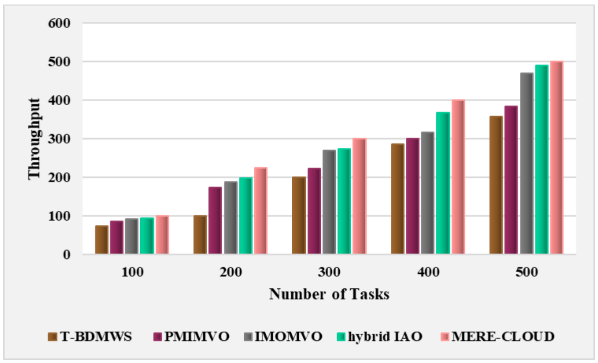

Figure. 5 Comparison of Throughput

The proposed MERE-CLOUD throughput analysis compared to other techniques is shown in Figure 5.

The proposed MERE-CLOUD technique that is being performs better in terms of throughput than others. As the number of total tasks rises, the throughput value for all techniques falls. The initial throughput number for node 100 obtained by the proposed method is 99%. From the analysis that the proposed MERE-CLOUD method has a faster data transmission rate. The performance of throughput decreases with the T-BDMWS [20].

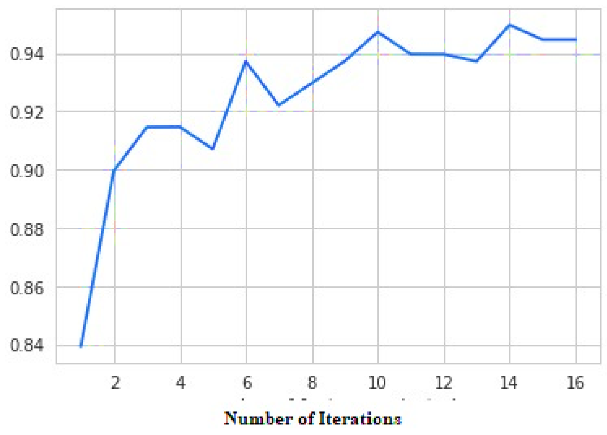

Figure. 6 Fitness function

The best solution is decided by the degree of performance of the pool of solutions obtained, which assesses the algorithm's efficiency. The method was analysed by examining the fitness function of the firefly algorithm as presented in Figure 6. The findings were based on the fitness function and the number of iterations it took to complete each function. The obtained results demonstrate that the algorithm is capable of delivering the best possible results as presented in Figure 7. This test distributes 100 jobs over four virtual machines on four cloud computing resource nodes, and calculates the average execution time for all tasks. The task duration arise as the number of tasks increased as presented in Figure 7, but it completed the optimal task time, indicating that the proposed approach works better in resource scheduling.

The analysis of the neural network techniques' comparative performance is shown in Figure 8. The proposed MERE-CLOUD model achieves 98.72% accuracy, 93.17% of sensitivity, 95.5% of recall and 96.72% of specificity when compared to prior neural network approaches. The existing ANN has low performance scores and the RNN and CNN approaches yielded results that were acceptable for each metric, although they were much different from the proposed model. This research demonstrates that the proposed approach results in high scores across all performance measures.

Figure 9 shows the task execution time in loose situation. It is found out that the proposed MERE-CLOUD approach has the minimum task execution time. The resource contentions occur when best-effort task is preempted by the task. As resource contention is less in loose situation, so that estimated finish time of task is close to the actual finish time. Hence existing techniques does not impact the job execution time significantly.

Received: March 11, 2024. Revised: May 31, 2024.

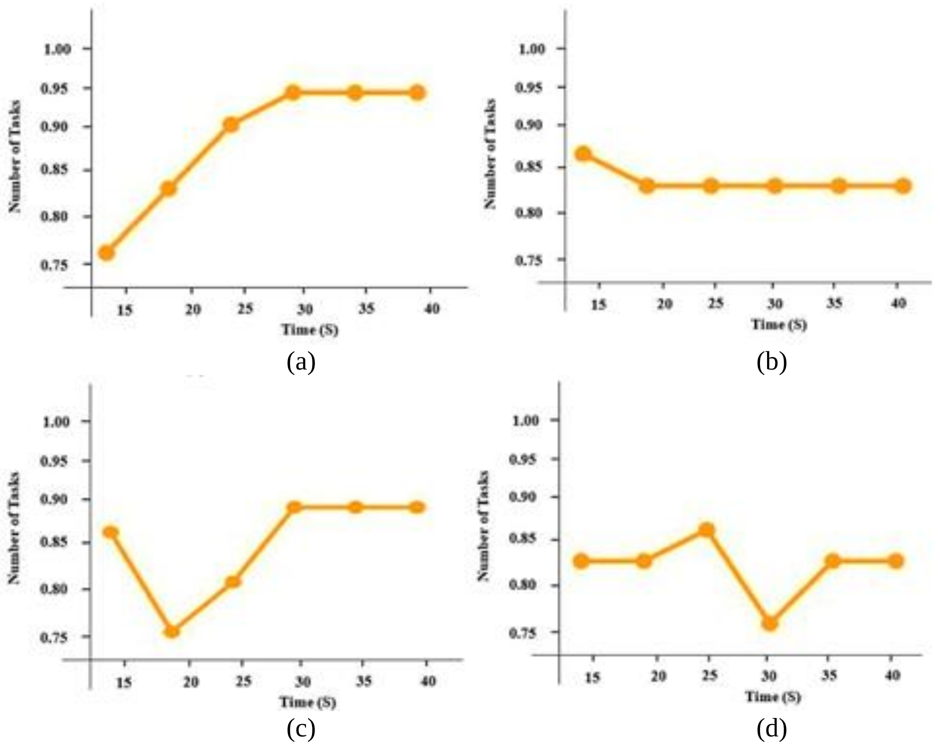

Figure. 7 Task Execution on different Cloud Resource Nodes: (a) Virtual Machine 1, (b) Virtual Machine 2, (c) Virtual Machine 3, and (d) Virtual Machine 4

5. Conclusion

Cloud computing's main goal is to provide cloud clients with the best possible big data task scheduling with the least amount of downtime, as well as load balancing and higher concurrency. The fundamental goal of big data job scheduling algorithms is to achieve these objectives as quickly as possible. Because job scheduling is such an important part of increasing the overall efficiency of complex resource allocation techniques, the proposed method addresses a wide range of task scheduling issues. Finally, the calculation results demonstrate that the proposed approach yields the best results, which may be 95.21% better than the top results obtained from the comparison algorithms in terms of failure rate. VM cost, network bandwidth, VM sort, RAM capacity, penalty, data center load, disc size, and CPU processing speed are the most important characteristics. Depending on these characteristics condition, an optimal algorithm it will be conferred in future work.

Conflicts of Interest

The authors declare that they have no conflicts of interest to report regarding the present study.

Author Contributions

The following statements should be used as follows: “Conceptualization, Visalaxi. G and Muthukumaravel. A; methodology, Visalaxi. G; software, Muthukumaravel. A; validation, Visalaxi. G and Muthukumaravel. A; formal analysis, Muthukumaravel. A; investigation, Muthukumaravel. A; resources, Visalaxi. G data curation, Muthukumaravel. A; writing—original draft preparation, Visalaxi. G writing—review and editing, Muthukumaravel. A; visualization, Visalaxi. G; supervision, Muthukumaravel. A; project administration, Visalaxi. G; funding acquisition, Muthukumaravel. A”, etc.

Acknowledgments

Authors should thank those who contributed to the article but cannot include themselves.

References

[1] X. Liu, J. Yu, J. Wang, and Y. Gao, “Resource allocation with edge computing in IoT networks via machine learning”, *IEEE Internet of Things Journal*, Vol. 7, No. 4, pp.3415-3426, 2020.

[2] D.K. Jain, S.K.S. Tyagi, S. Neelakandan, M. Prakash, and L. Natrayan, “Metaheuristic optimization-based resource allocation

technique for cybertwin-driven 6G on IoE environment", *IEEE Transactions on Industrial Informatics*, Vol. 18, No. 7, pp. 4884-4892, 2021.

[3] A.M. Seid, G.O. Boateng, S. Anokye, T. Kwantwi, G. Sun, and G. Liu, "Collaborative computation offloading and resource allocation in multi-UAV-assisted IoT networks: A deep reinforcement learning approach", *IEEE Internet of Things Journal*, Vol. 8, No. 15, pp. 12203-12218, 2021.

[4] J. Praveenchandar, and A. Tamilarasi, "Dynamic resource allocation with optimized task scheduling and improved power management in cloud computing", *Journal of Ambient Intelligence and Humanized Computing*, Vol. 12, pp. 4147-4159, 2021.

[5] K. Raghavendar, I. Batra, and A. Malik, "A robust resource allocation model for optimizing data skew and consumption rate in cloud-based IoT environments", *Decision Analytics Journal*, Vol. 7, pp. 100200, 2023.

[6] S.K. Chowdhary, and A.L.N. Rao, "QoS Enhancement in Cloud-IoT Framework for Educational Institution with Task Allocation and Scheduling with Task-VM Matching Approach", *Wireless Personal Communications*, Vol. 121, pp. 267-286, 2021.

[7] S.B. Sangeetha, R. Sabitha, B. Dhiyanesh, G. Kiruthiga, N. Yuvaraj, and R.A. Raja, "Resource management framework using deep neural networks in multi-cloud environment.Operationalizing Multi-Cloud Environments: Technologies", *Tools and Use Cases*, pp. 89-104, 2022.

[8] S. Abedi, M. Ghobaei-Arani, E. Khorami, and M. Mojarad, "Dynamic resource allocation using improved firefly optimization algorithm in cloud environment", *Applied Artificial Intelligence*, Vol. 36, No. 1, pp. 2055394, 2022.

[9] Y. Xu, and A.H. Mohammed, "An energy-aware resource management method in cloud-based Internet of Things using a multi-objective algorithm and crowding distance", *Transactions on Emerging Telecommunications Technologies*, Vol. 34, No. 1, pp. e4673, 2023.

[10] R. Aron, and D.K. Aggarwal, "Resource scheduling of concurrency-based applications in IoT based cloud environment", *Journal of Ambient Intelligence and Humanized Computing*, Vol. 14, No. 6, pp. 6817-6828, 2023.

[11] T. Salehnia, A. Seyfollahi, S. Raziani, A. Noori, A. Ghaffari, A.R. Alsoud, and L. Abualigah, "An optimal task scheduling method in IoT-Fog-Cloud network using multi-objective moth-flame algorithm", *Multimedia Tools and Applications*, pp.1-22, 2023.

[12] F. Arvaneh, F. Zarafshan, and A. Karimi, "Using fog computing (FC) and optimization techniques for tasks migration and resource allocation in the internet of things (IoT)", *International Journal of Computers and Applications*, pp. 1-9, 2024.

[13] Q. Huangpeng, and R.O. Yahya, "Distributed IoT services placement in fog environment using optimization-based evolutionary approaches", *Expert Systems with Applications*, Vol. 237, pp. 121501, 2024.

[14] P. D. Kusuma, and A. Dinimaharawati, "Swarm Bipolar Algorithm: A Metaheuristic Based on Polarization of Two Equal Size Sub Swarms", *International Journal of Intelligent Engineering and Systems*, Vol. 17, No. 2, 2024, doi: 10.22266/ijies2024.0430.31.

[15] P. D. Kusuma, and M. Kallista, "Swarm Space Hopping Algorithm: A Swarm-based Stochastic Optimizer Enriched with Half Space Hopping Search", *International Journal of Intelligent Engineering and Systems*, Vol. 17, No. 2, 2024, doi: 10.22266/ijies2024.0430.54.

[16] P. Kusuma, and A. L. Prasasti, "Walk-Spread Algorithm: A Fast and Superior Stochastic Optimization", *International Journal of Intelligent Engineering and Systems*, Vol. 16, No. 5, pp. 275-288, 2023, doi: 10.22266/ijies2023.1031.24.

[17] P. D. Kusuma, and F. C. Hasibuan, "Attack Leave Optimizer: A New Metaheuristic that Focuses on The Guided Search and Performs Random Search as Alternative", *International Journal of Intelligent Engineering and Systems*, Vol. 16, No. 3, pp. 244–257, 2023, doi: 10.22266/ijies2023.0630.19.

[18] K.K. Chakravarthi, and L. Shyamala, "TOPSIS inspired budget and deadline aware multi-workflow scheduling for cloud computing", *Journal of Systems Architecture*, Vol. 114, pp. 101916, 2021.

[19] M. Otair, A. Alhmoud, H. Jia, M. Altalhi, A.M. Hussein, and L. Abualigah, "Optimized task scheduling in cloud computing using improved multi-verse optimizer", *Cluster Computing*, Vol. 25, No. 6, pp. 4221-4232, 2022.

[20] L. Abualigah, M.A. Elaziz, N. Khodadadi, A. Forestiero, H. Jia, and A.H. Gandomi, "Aquila optimizer based PSO swarm intelligence for IoT task scheduling application in cloud computing", *Integrating Meta-heuristics and Machine Learning for Real-world Optimization Problems*, pp. 481-497, 2022.

[21] A. Maroosi, and R.C. Muniyandi, “A novel membrane-inspired multiverse optimizer algorithm for quality of service-aware cloud web service composition with service level agreements”, *International Journal of Communication Systems*, pp. e5483, 2023.

[22] P. D. Kusuma, and A. Dinimaharawati, “Extended Stochastic Coati Optimizer”, *International Journal of Intelligent Engineering and Systems*, Vol. 16, No. 3, 2023, doi: 10.22266/ijies2023.0630.38.

[23] P. Kusuma, and A. Dinimaharawati, “Four Directed Search Algorithm: A New Optimization Method and Its Hyper Strategy Investigation”, *International Journal of Intelligent Engineering and Systems*, Vol. 16, No. 5, pp. 598-611, 2023, doi: 10.22266/ijies2023.1031.51.

[24] P. D. Kusuma, and A. Novianty, “Total Interaction Algorithm: A Metaheuristic in Which Each Agent Interacts with All Other Agents”, *International Journal of Intelligent Engineering and Systems*, Vol. 16, No. 1, pp. 224-234, 2023, doi: 10.22266/ijies2023.0228.20.

[25] P. D. Kusuma, and M. Kallista, “Migration-Crossover Algorithm: A Swarm-based Metaheuristic Enriched with Crossover Technique and Unbalanced Neighbourhood Search”, *International Journal of Intelligent Engineering and Systems*, Vol. 17, No. 1, 2024, doi: 10.22266/ijies2024.0229.59.