Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X
IMPACT FACTOR: 7.056

IJCSMC, Vol. 15, Issue. 3, March 2026, pg.21 – 34

Logistic Migration Resource Shortest Distribution using Fairness Flashing Firefly Approach for Efficient Task Scheduling and Resource Allocation in the Cloud Environment

S. Dharmaraj[1]; P. Kavitha[*2]

[1]Head & Assistant Professor, Department of Computer Applications, Loyola College, Vettavalam, India
[2]Department of Mathematics, Amrita School of Physical Sciences, Coimbatore, Amrita Vishwa Vidyapeetham, India
Corresponding Author: P. Kavitha
[1] dharmaraj@lcv.edu.in; [2*] p_kavitha@cb.amrita.edu

DOI: https://doi.org/10.47760/ijcsmc.2026.v15i03.003

Abstract: Cloud Computing (CC) is a large-scale platform supporting a variety of services, including computing, storage, and analysis over the Internet. It provides greater agility, scalability, and flexibility to users and organizations. The rapid growth of cloud users presents challenges in efficiently scheduling demanding workloads across available resources. Therefore, Task Scheduling (TS) is a key issue when establishing a CC system. The TS problem is crucial in CC environments, as the performance of the cloud relies primarily on efficient task scheduling. Also, it is a dynamic environment, and allocating resources efficiently and accurately can be a difficult process. Most of the previous methods don't meet efficient Resource Allocation (RA). Motivated by this, the paper presents the Logistic Migration Resource Shortest Distribution (LMRSD) approach based on the Fairness Flashing Firefly Approach (F3A) for effective task preparation and Cloud Environment (CE) in optimal resource. In this approach, the Minimum Maximum Task Processing Rate (M2TPR) method is employed to split the incoming task in the cloud. Based on the divided tasks fed into the Maximum Correlation Ranking analytics (MCRA), the algorithm utilizes ordering the tasks. Afterward, the proposed Fairness Flashing Firefly Approach (F3A) is used to schedule the task from ordered jobs. Finally, the LMRSD technique proficiently assigns the scheduled tasks to the Virtual Machines (VMs) with least response time in the CE. Thus, the proposed approach improves the efficiency of scheduling and resource scheduling performance with the least migration time performance than conventional methods.

Keywords: Cloud Computing, Job, Task Scheduling, Resource Allocation, Simulated Machines, Rank.

I. Introduction

Cloud computing (CC) is a versatile technology that can be used for Data Preservation, Statistical Analysis, Internet of Things (IoT) applications, and software development [Gawali, M.B et al. (2018) and T. Aladwani et al. (2020)]. It offers a range of benefits for businesses, including increased efficiency, flexibility, and cost savings. The payment model of cloud computing attracts companies and individuals to run applications on the cloud [Anupama K Channappa et al. (2022)]. Cloud applications come in varying sizes, resource requirements, and execution times. They are separated into individual executable files known as jobs, tasks, and events. Therefore, Task Scheduling (TS) and Resource Allocation (RA) in CC involve assigning resources like CPU, memory, and storage to tasks on the CE.

As user business needs and quality of service requirements continue to increase, the system performance is facing significant challenges and complicated, leading to task scheduling failures [Ergu, D et al. (2013) and Lei Zhang et al. (2017)]. Efficient TS and RA of cloud resources are a crucial issue that needs to be handled due to the heterogeneity, interdependence, and unpredictability of resources in a CE [Rajni Aron et al. (2022)]. Thus, TS is the process of coordinating incoming requests (jobs) to ensure proper utilization of available resources. Each service has many users, so many requests (jobs) can be created simultaneously. TS is an important concern in the realm of cloud calculation. Effective preparation of tasks results in optimal utilization of resources. Figure 1 depicts the TS and RA working process

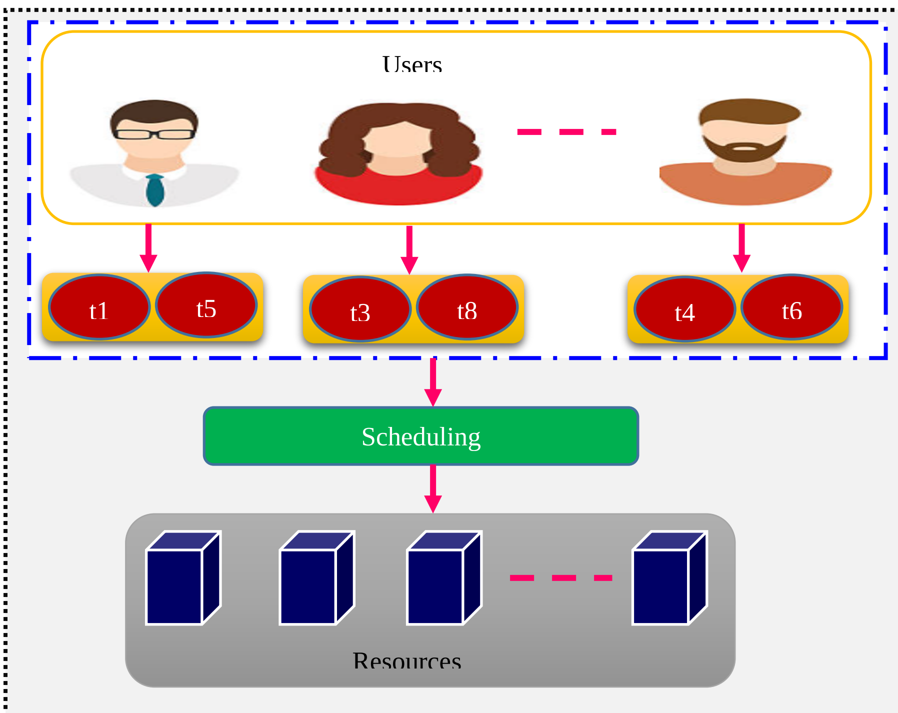

Fig. 1. TA and RA process in cloud

This research work aims to automate workload scheduling in cloud environments using VM technology based on varying task requirements. The main aim is to enhance task scheduling and resource allocation using F3A and MCRA techniques. This proposed work focusses to reduce migration time and improve scheduling performance and resource utilization performance.

II. Related Work

Zheyi Chen et al. (2020), In order to offload neural network components to distant clouds with adequate resources, the author implemented DNN-based apps in CC. The difficulty of this unloading procedure generates excessive delays and negatively impacts the user experience

Daqin Wu et al. (2018), the author carried out a TS approach that combats the current inefficiencies using a PSO algorithm. This method's drawback is that its scheduling strategy is only effective for local performance; it is unable to meet the demands of global performance at high time costs. TS time costs can be decreased because of the superior convergence effect.

Yanyue Yu et al. (2019), the author carried out a cloud the three queues implemented in TS method and dynamic priorities to understand how cloud computing clusters are becoming more diverse and are experiencing significant differences in system loads.

PeiYun Zhang et al. (2018), the author carried out a cloud the TS framework is based on a two-step strategy and preemptively installs virtual machines based on historical scheduling data, saving time waiting for virtual machines to be created. The challenge of this method is dynamically match tasks to the best virtual machines to save execution costs.

Yonghua Xiong et al. (2019), the author carried out a relating Johnson's Law to Genetic Algorithms, was established that took into account the CDC multiprocessor scheduling characteristics. The challenge of the method is the performance of JRGA, list scheduling algorithm and compared the improved list scheduling algorithms through simulation.

Belal Ali Al-Maytami et al. (2019), the author carried out a novel scheduling algorithm of PTCT based DAG evaluates the best scheduling algorithm for sensitive cloud data. The challenge of this method is by employing Principal Component Analysis and reducing the ETC matrix, it provides significant improvements in makes pan, reducing computational effort and complexity.

Kun Guo et al. (2020), The Resource Constrained TS Profit Optimization algorithm, which includes clustering preprocessing, classification, profit matrix generation, and optimal scheduling policy calculation, was implemented by the author. The problem with this approach is that the number of clouds is steadily growing, so efficient job scheduling techniques are needed to maximize the advantages of edge clouds.

Haiyu Zhang et al. (2023), The Multi Objective TS Model, which has a big impact on a lot of applications, was studied by the author. It can guarantee system stability and enhance multi-objective scheduling in conventional cloud computing. Nevertheless, the inefficiency and single-objective optimization limitations of the conventional cloud computing TS approaches prevent them from effectively meeting the demands of cloud computing workloads.

Shanchen Pang et al. (2019), In order to improve load balancing, the author used an EDA-GA Hybrid Scheduling algorithm. The problem with this approach is that it takes longer to load the assignment when multiple cloud users access the search engine simultaneously.

Kaixuan Kang et al. (2021), the author proposed an energy-efficient cloud computing TS architecture based on adaptive deep reinforcement learning. The challenge of this approach is that the operational requirements of complex cloud data centers are extremely high. It is necessary to ensure that resources are available in an energy-saving manner and to meet the different specification of approach. Table 1 indicates the evaluation algorithms of task scheduling

TABLE I
EVALUATION OF ALGORITHMS FOR TASK SCHEDULING

Author/Year	Techniques Used	Advantages	Limitations/Drawbacks
Z. Liu, M. Liwang et al. (2023)	RFID	It has the highest priority for vehicle dispatching, can quickly execute tasks and provide reliable transmission links.	Different type of tasks requires various types of resources, some tasks required large type of storage.
H. Yuan et al. (2022)	T.S. method	Energy costs and ATLP are low compared to other commonly available products.	The performance shows the analysis only when occurs runtime scheduling process.
L. Zhao et al. (2020)	New resource allocation algorithm (HPS+)	Employ a modified hypergraph partitioning technique to minimize traffic.	High data volumes and slow transfer speeds.
H. Yuan et al. (2021)	Temporal Task Scheduling (TTS)	Temporal changes in green hybrid clouds and rank all task within latency bounds.	The nature of the tasks presented significant challenges in scheduling tasks for each application given the GDC's limited infrastructure resources
S. Mousavi et al. (2023)	DNSGA-II	By using the new operator, can control selection pressures on agents and balance their exploration and exploitation abilities.	System performance is low due to poor quality of service and energy consumption of computing equipment.

Pallab Banerjee et al. (2023), the author describes the Dynamic Heuristic Johnson Sequencing Technology complete the optimal sequence of data on each support, productively minimize the completion time. The challenge of this method in this method shows excellent scalability and efficiency in solving complex JS in cc environments.

Hadeer Mahmoud et al. (2022), the author describes a Novel Task Scheduling Decision Tree method for assigning and perform application tasks to reducing make span and improved load balancing. The challenge of this method it refers to the method of rationally ordering and allocating user-specified application tasks to run on virtual machines.

Yanal Alahmad et al. (2021), the author described a Fault Aware TS Framework that predicts the exit status of specific tasks at runtime and takes appropriate corrective actions, minimize the failure probability of tasks and resources usage. The challenge of this method the focus is on predicting task failure and neglecting remedial actions.

Xuan Chen et al. (2020), The author implemented an advanced method known as C Loud Task Scheduling with improved WOA in order to enhance the classification's facility to search for optimal resolutions. One of the main challenges of this method is that the task preparation in cloud computing can have a direct impact on resource utilization and operational costs.

Swati Lipsa et al. (2023), the author described Prioritization method is designed to prioritize incoming tasks using an advanced data structure known as delay matrix to allocate tasks with the highest priority. The challenge of this method is a TS problem for the cc environment is built using the M/M/n queuing model.

Shaojin Geng et al. (2020), the author carried out a cloud model with many objectives, minimize time and cost, maximize source allocation and distribute load. At the same apparency, the Hybrid Angle-Based Multi-Objective Optimization Algorithm solves this model. The challenge of this method the quality of dispatch performance directly impacts customer satisfaction and provider profits.

Deafallah Alsadie et al. (2021), the author carried out a MOFWOTS is to identify Near Optimal TS solutions while handling competing goals. The challenge of this method Optimize system performance by optimally mapping cloud tasks to resources to meet the compute and storage needs of high-performance applications while meeting CS goals such as performance, completion time, and resource utilization.

Jin Sun et al. (2020), the author carried out an efficient planning algorithm that solves MOO problems and it can generate a Pareto Optimal solution. It provides a balance for Algorithmic Optimization and Energy Conservation. The challenge of this method existing cloud solutions generally suffers from low resource utilization when processing large-scale remote sensing facts sets, affects the value of facility of cc.

E Cao et al. (2022), the author carried out to solve this problem Adaptive Power Management, a new task planning method is dedicated to cost optimization based on genetic algorithms. The challenge facing the method is that excess cloud service providers are introducing new pricing strategies that consider users pay based on different combinations of assigned CPU frequency and virtual machine architecture and pricing.

Qi Fan et al. (2018), the author carried out a method of order prioritization based on Heterogeneous Earliest Completion Time Based on Togetherness to the Ideal Solution is called the HEFT-T algorithm. The challenge of this project, it becomes difficult to schedule actions on user-submitted tasks and optimize multiple objectives while meeting user needs. Table 2: shows the instant of papers linked to cloud task preparation.

TABLE II ANALYSIS OF PAPERS RELATED TO CLOUD TASK SCHEDULING

Ref.	Year	Applied Algorithm	Parameters	Findings
F. A. Saif et al.	2023	MGWO	Transmission delay, Energy consumption, Processing data.	The performance of MGWO algorithm is minimizing energy consumption than previous methods.
K. Li et al.	2022	PSOMC	Task scheduling time, consumption cost	The PSOMC algorithm performance is slightly better than the ACO method
M. Zorzi et al.	2020	Two-Timescale Framework	Power consumption, task latency	In comparison to the other two techniques, the two-time scale architecture can give more reliable task latency assurances under varying workload density conditions.
B. Saemi et al.	2023	HMHHO	Processing power, complex tasks, extensive search techniques	The Hybrid Multi Objective Harris-Hawks Optimization approach completes the task faster and consumes less energy on average than the GA, ACO, PSO, and CSA methods.
J. Xuan et al.	2020	Cloud Adaptive Genetic Algorithm	Fuzzy environment, flexible operations	In the JS, the CAG method improves the efficiency of iterative search compared with other algorithms.

III. PROPOSED METHODOLOGY

Figure 2 exhibits a proposed overall architecture diagram for efficient TS and RA in a Cloud Environment (CE). In the Data Centers (DC) executed, different types and sizes of tasks (jobs) arrive from cloud users. In the first stage, the Minimum Maximum Task Processing Rate (M2TPR) method is used for splitting the tasks based on similar and dissimilar types of jobs. Then, the tasks in the work ranking are strictly sequenced according to a Maximum Correlation Ranking Analytics (MCRA) approach. In the next stage, the F3A technique is used to schedule the ranking tasks. Then, LMRSD continuously observes the VM load.

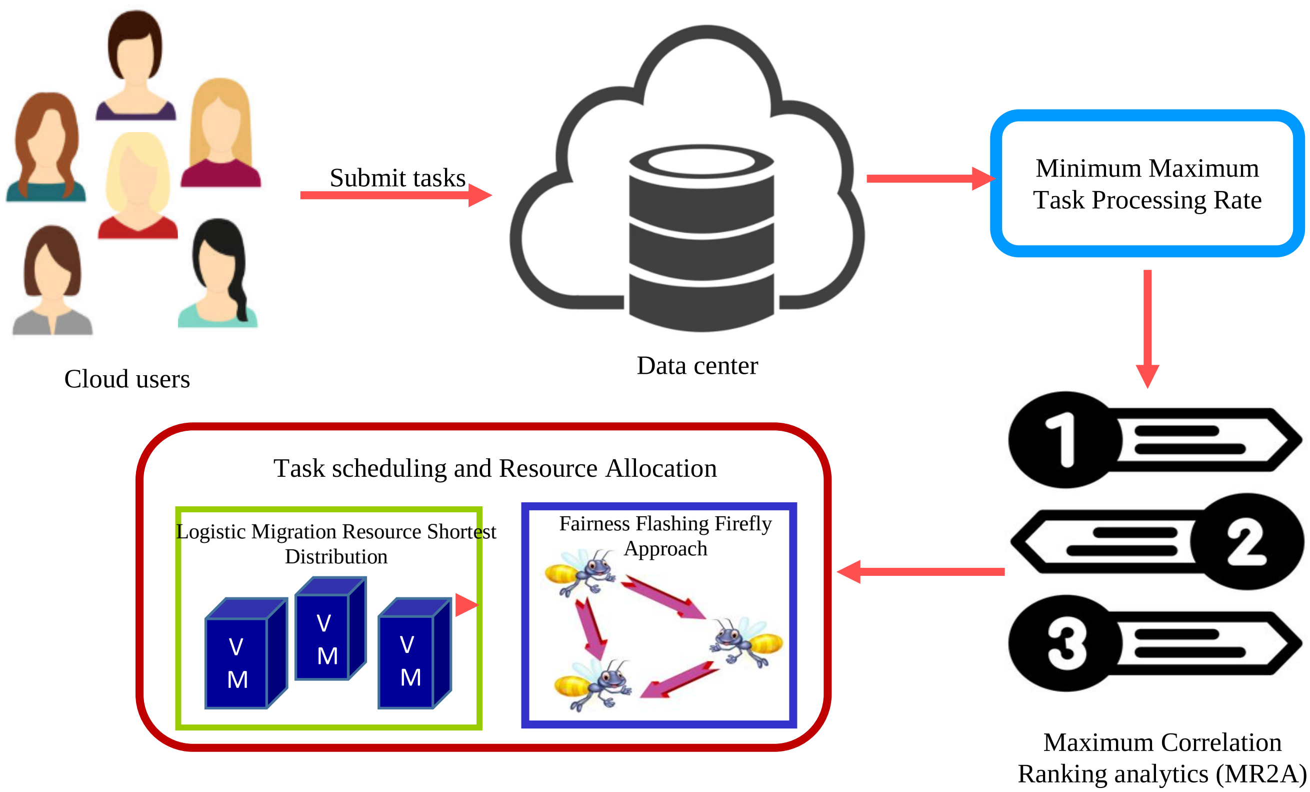

Fig. 2. Proposed architecture diagram

Moreover, current grade of other VMs. In that scenario, if the existing VM is loaded, but other VMs are inactive, they will be maintained. Once this is discovered, the recommended approach divides up tasks and allocates them to other virtual computers.

A. *Maximum Correlation Ranking analytics*

In this section, we utilize the Maximum Correlation Ranking Analytics (MCRA) method to sequence tasks according to their length and job execution time. Table 1 describes the numerical result for task preference.

TABLE III
TASK RANKING PREFERENCES MODEL

Numerical values	Preference
1	Equally desired
2	Moderately desired
3	Intensely desired

4	Favorably desired
5	Highly desired

The table above shows numerical values for the ranking preference model based on task length and length. The recommended approach prioritizes processes based on their length and execution time

TABLE IV
MCRA ALGORITHM FOR RANKING PROCESS

Rank	Before Ranking order	After ranking order
1	T3	T8
2	T6	T3
3	T8	T21
4	T21	T9
5	T9	T6

Table 2 explains the before ranking and after ranking order based on the length of the task from the divided tasks.

A. *Fairness Flashing Firefly Approach (F3A)*

The proposed F3A method is a nature-inspired process that mimics the behavior and attraction patterns of fireflies, using flashing lights as a basis. All fireflies are unisex, allowing them to attract others without regard to gender. Flies' can be identified based on the intensity of the FL used for communication (i.e., TS in a cloud environment). The attraction of fireflies is related to the strength of task scheduling from ranking jobs, and the fewer distance between the fireflies determines this strength. As the distance increases, the fairness attraction decreases. Fireflies with lower flashing are attracted to brighter ones. If a fly is the brightest, it moves randomly for task scheduling. Therefore, this proposed approach must estimate the brightness of the firefly, which encompasses intensity and fairness flash attractiveness for effective task analysis in the cloud environment. The F3A approach workflow diagram for specific task analysis is presented in figure 3.

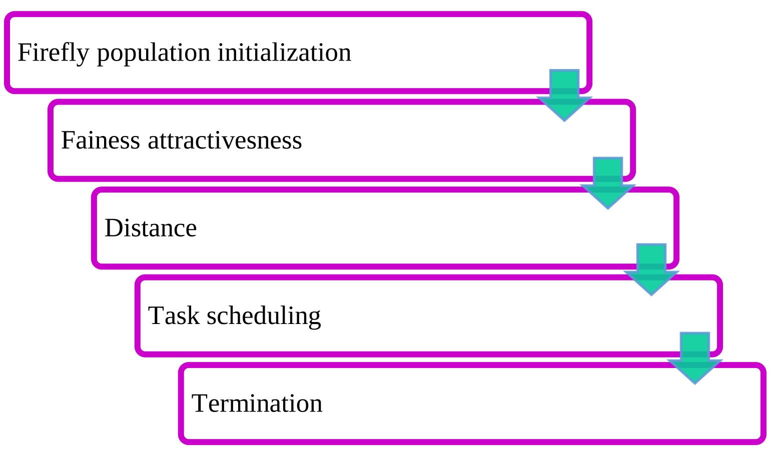

Fig. 3. F3A approach work process for task scheduling

Here, mathematically, the F3A technique estimates the following equations.

$$L_{in} = \mathcal{F}_{\mathcal{L}} X exp^{-\alpha \mathfrak{D}_m^2}$$

The above equation find the light intensity of fireflies. Wherein, $\mathcal{F}_{\mathcal{L}}$ denotes initial flash brightness values of every fireflies, α is a coefficient of light absorption with distance among any two fireflies m and$n$. Then, the estimate of the fairness attractiveness of the fireflies is followed by the below equation,

$\beta(\alpha) = \beta_0 X exp^{-\alpha \mathfrak{D}_m^2}$

Here, β_0 is an initial value of fairness attractive fireflies. Two fireflies distance $\mathfrak{D}$ is estimated in below equation,

$\mathfrak{D} = \sqrt{\sum_{i=1}^{\mathfrak{d}} (p_{mE} - p_{nE})^2}$

Let assume that, distance among two fireflies m and n their positionsp_m, p_n respectively, $\mathfrak{d}$ is a dimension, and E is the element.

Algorithm 1. Task scheduling

Begin

Randomly generate the fireflies $\mathfrak{f}_i (i = 1,2, \dots N)$

Estimate Light intensity of each fireflies

$While(i < M_G)$ // M_G denotes maximum generation

$For\ each\ i = 0\ to\ N$ // N denotes number of fireflies

Check fairness firefly attractiveness

$If(\mathfrak{f}_m < \mathfrak{f}_n)$

Compute the two firefly distance

Move the firefly "m" closer to "n"

$$p_m(\mathcal{G} + 1) = p_m(\mathcal{G}) + \beta_0 X exp^{-\alpha \mathfrak{D}_m^2} X\left(p_n(\mathcal{G}) - p_m(\mathcal{G})\right) + \gamma X(\mathbb{R}\mathfrak{v} - 0.5)$$

End if

Update their positions

Compute the task weight $\mathfrak{W}$

$$\mathfrak{W} = \frac{\mathbb{D}_t - \mathbb{S}_t}{\mathbb{E}_t}$$

End for

End while

Return ←Task scheduled.

Stop

Algorithm 1 explains the task scheduling based on fireflies' natural behavior. In this mechanism, if one firefly is more attractive than another firefly, it moves towards the more attractive firefly. Then, weight values aid in prioritizing tasks with imminent deadlines. Every task is estimated by weight and prepared to allot to the suitable VM for the process. With this, our tasks are easily scheduled in the cloud. Here, $\mathcal{G}$ is a generation, γ denotes positive parameters, $\mathbb{R}\mathfrak{v}$ present random number (i.e. [0 1]),$\mathbb{D}_t$ is a task deadline, $\mathbb{S}_t$ denotes task start time and$\mathbb{E}_t$ is a task execution time. Figure 4 explains the flow chart for task scheduling in the cloud environment.

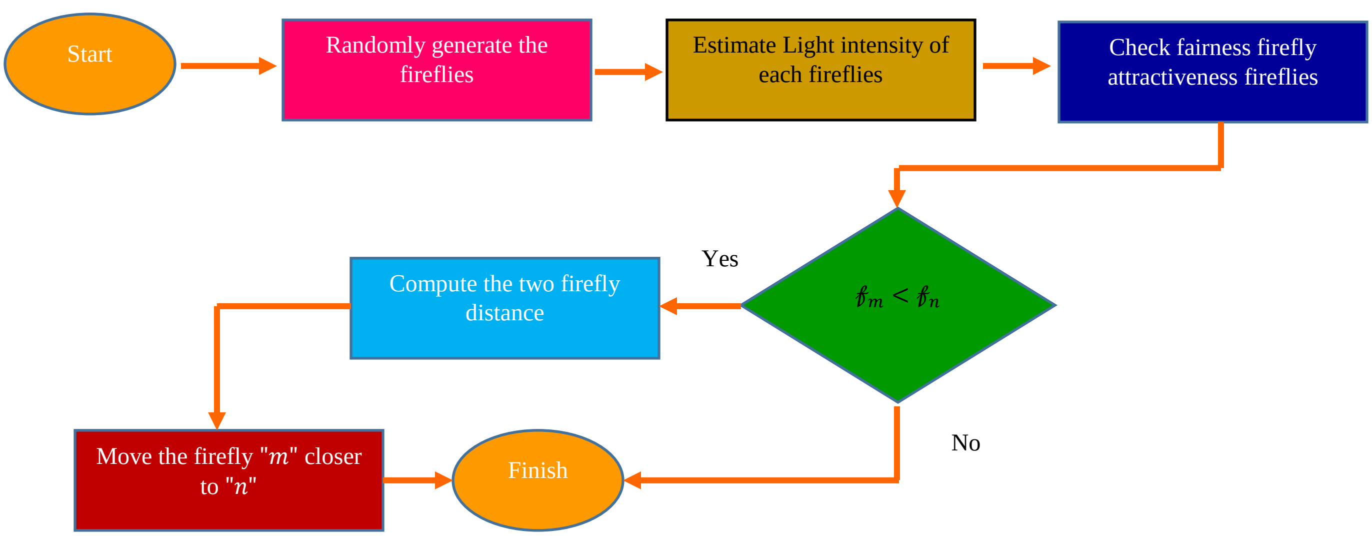

Fig. 4. Flowchart for task scheduling in cloud

B. *Logistic Migration Resource Shortest Distribution*

After scheduling the tasks, we use the LMRSD method for resource allocation to appropriate VMs in the cloud. Thus, the proposed algorithm checks which VMs are busy or not. Because when jobs are assigned to VMs, if that VM is overloaded, our proposed method better allocates work to idle VMs. Let us assuming that, scheduled tasks are small$\mathcal{S}_t$, on-demand memory$\mathcal{O}_m$, compute intensive $\mathcal{C}_I$and large tasks$\mathcal{L}_t$.

$\mathcal{S}_t = \{t_1^{\mathcal{S}}, t_1^{\mathcal{S}} \ldots \ldots t_N^{\mathcal{S}}\}$

$\mathcal{O}_m = \{t_1^{\mathcal{O}_m}, t_1^{\mathcal{O}_m} \ldots \ldots t_N^{\mathcal{O}_m}\}$

$\mathcal{C}_I = \{t_1^{\mathcal{C}_I}, t_1^{\mathcal{C}_I} \ldots \ldots t_N^{\mathcal{C}_I}\}$

$\mathcal{L}_t = t_1^{\mathcal{L}_t}, t_1^{\mathcal{L}_t} \ldots \ldots t_N^{\mathcal{L}_t}$

The above equation is the list of scheduled tasks in the cloud environment. During resource allocation, the cloud system configures virtual machines that meet each task's resource requirements. Therefore, the cost of a task fluctuates according to its resource needs. This method focuses on minimizing the cost function, which is evaluated by calculation the CPU and recollection cost functions of the VM.

$C_F = \sum_{i=1} A_{cost}(i)$

$M_F = \sum_{i=1} A_{memory}(i)$

The above equation is used to analysis the cost function of VM's CPU cost and memory. Let us assuming that, CPU associated cost $A_{cost}$and memory A_{memory} are estimated in below equations,

$A_{cost}(i) = b_{cost} \times C_{VM} \times R_{dkj} + T_c$

Here b_{cost} (0.20/hr) and T_c (0.005) are constant.

$A_{memory}(i) = bc_M \times M_{VM} \times P_{Tkj} + M_{tc}$

Here, bc_M $0.10/GB/hr$ and $M_{tc} = 0.80$ are constant.

Where,

k – VM
j – Tasks
b_{cost} – Base cost
C_{VM} – CPU of VM
R_d – Task run time duration
T_c – Transaction
bc_M – Memory base cost
M_{VM} – Virtual Machine memory
P_T – processing Time
M_{tc} –Transmission cost memory

If each task is assigned to a VM, the determined time taken by a virtual appliance to execute all tasks is known as makes pan m_s, as indicated in the below equation.

$m_s = maximum \sum_{k=1}^{N} \mathfrak{C}_{Tkj}$

Wherein, $\mathfrak{C}_T$ denotes overall task completion time and N implies the total number of tasks.

Algorithm 2: Resource allocation

Input: Set of scheduled tasks t and VM
Output: Task allocation

Begin
 Initialize the assign of data $t = \{t_1, t_2 \ldots t_N\}$
 Initialize the assign of VM $VM = \{VM_1, VM_2 \ldots VM_N\}$
 For each $t: N$
  For each $VM: N$
  Compute VM maximum load L_{rate}
  $L_{rate}(T) = \frac{VM_{\mathcal{R}}(T)}{VM_{maximumL}} \times 100$ // T denotes time and $VM_{\mathcal{R}}$ denotes running time
   $If\ t \in \mathcal{O}_m$ Then
    Allocate task $t_j^{\mathcal{O}_m} \to VM_k^{\mathcal{O}_m}$
   $Else\ If\ t \in \mathcal{S}_t$ Then
    Allocate task $t_j^{\mathcal{S}_t} \to VM_k^{\mathcal{S}_t}$
   $Else\ If\ t \in \mathcal{L}_t$ Then

Allocate task $t_j^{\mathcal{L}t} \rightarrow VM_k^{\mathcal{L}t}$

$Else\ If\ t \in \mathcal{C}_{\mathrm{I}}$ Then

Allocate task $t_j^{\mathcal{C}_{\mathrm{I}}} \rightarrow VM_k^{\mathcal{C}_{\mathrm{I}}}$

End if

End for each

End for each

Return← efficient resource allocation

Stop

Algorithm 2 explains the appropriate source allocation grounded on scheduled tasks in the cloud calculation atmosphere. Here, $VM_k^{\mathcal{O}_m}$ is a VM's on-demand memory, $VM_k^{\mathcal{S}_t}$ is a VM's small tasks, $VM_k^{\mathcal{C}_{\mathrm{I}}}$ implies VM's compute intensive, $VM_k^{\mathcal{L}t}$ implies the large tasks and $VM_{maximumL}$ is a maximum load on virtual machine. Figure 5 describe the detailed process of Task development and resource allocation in CC

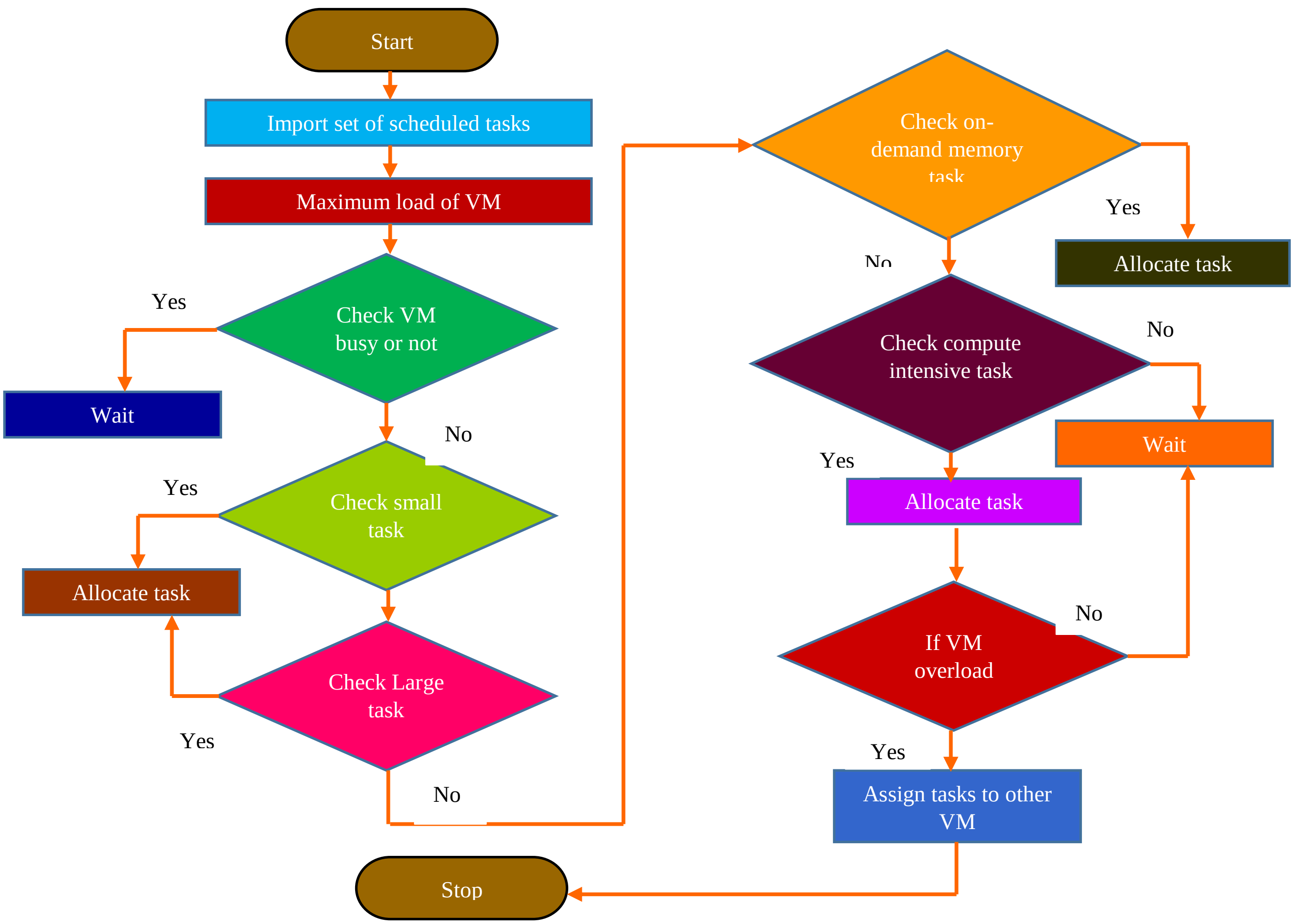

Fig. 5. Flowchart for Resource distribution in cloud computing

IV. EXPERIMENTAL RESULT ANALYSIS

This simulation verifies the benefits of the proposed framework in processes of Makspan, forecast, and resource utilization performance in a CC atmosphere.

TABLE V
EXPERIMENTAL SPECIFICATIONS

Simulation parameters	Description
Code	C#
Platform	Visual Studio
Number of tasks	100-500
Length of task	1500-4000
Data Center (DC)	1
Virtual Machine (VM)	10
Storage	500GB

The proposed approach for TS is tested using C# language with an I7 processor running at 2.20GHz, 16GB RAM, and Windows 10 operating approach, are presented in the table.

A. Parameter evaluation

Simulation parameters are resource utilization, scheduling performance, migration time, and time complexity (makespan).

TABLE VI
COMPARISON OF SCHEDULING PERFORMANCE

Performance in %					
Number of VM	Number of task	GA	RFID	HMOBA	F3A-LMRSD
10	100	67.59	70.42	73.68	81.24
	200	71.22	74.05	77.13	84.64
	300	73.47	78.12	79.49	88.71
	400	76.21	81.20	83.24	91.58
	500	79.08	82.16	86.13	95.27

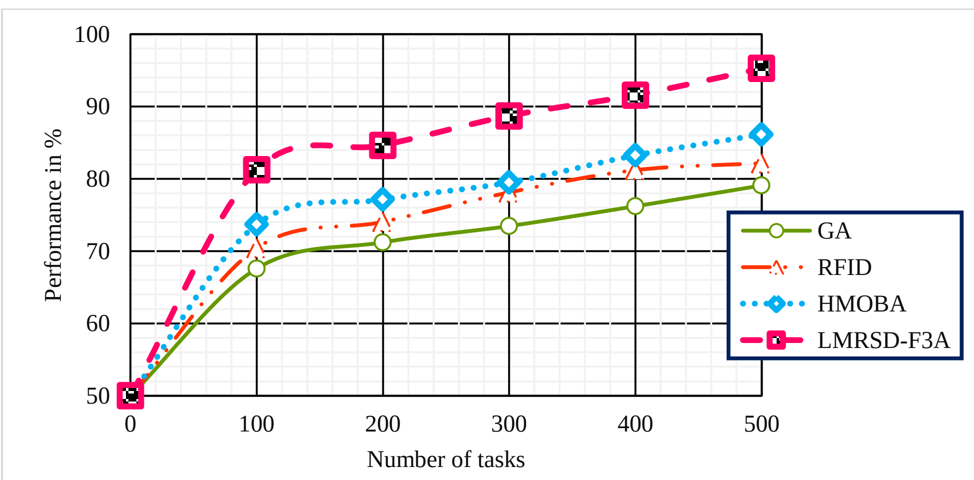

Fig.6. Analysis of scheduling performance

Table and Figure demonstrate the comparison of scheduling performance among the proposed and other methods. The simulation results indicate that the proposed processes outshines other methods in all aspects. For example, for 500 tasks, the F3A-LMRSD method achieved a 95.27% scheduling performance, while conventional methods like GA, RFID, and HMOBA achieved 79.08%, 82.16%, and 86.13%, respectively.

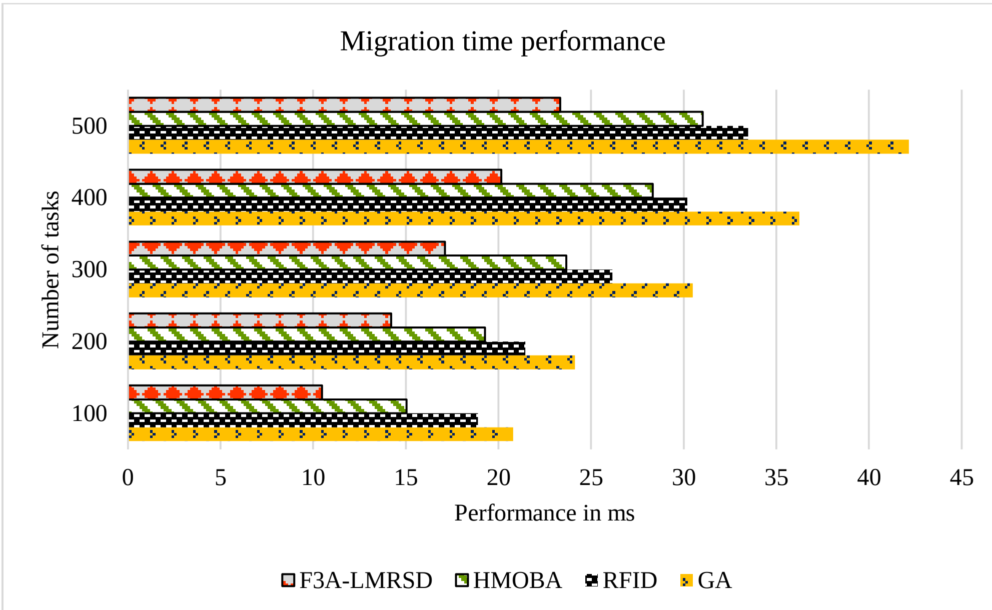

Fig.7. Comparison of migration performances

The figure demonstrates the graphic representation of the milliseconds for migration time performance using the proposed and different methods. It illustrates that the proposed F3A-LMRSD algorithm produces less time for resource allocation in the VMs. For instance, 500 tasks have 23.33^{2x}ms, while the existing methods are GA, RFID, and HMOBA obtained 42.16^{2x}ms, 33.48^{2x}ms, and 31.02^{2x}ms, respectively.

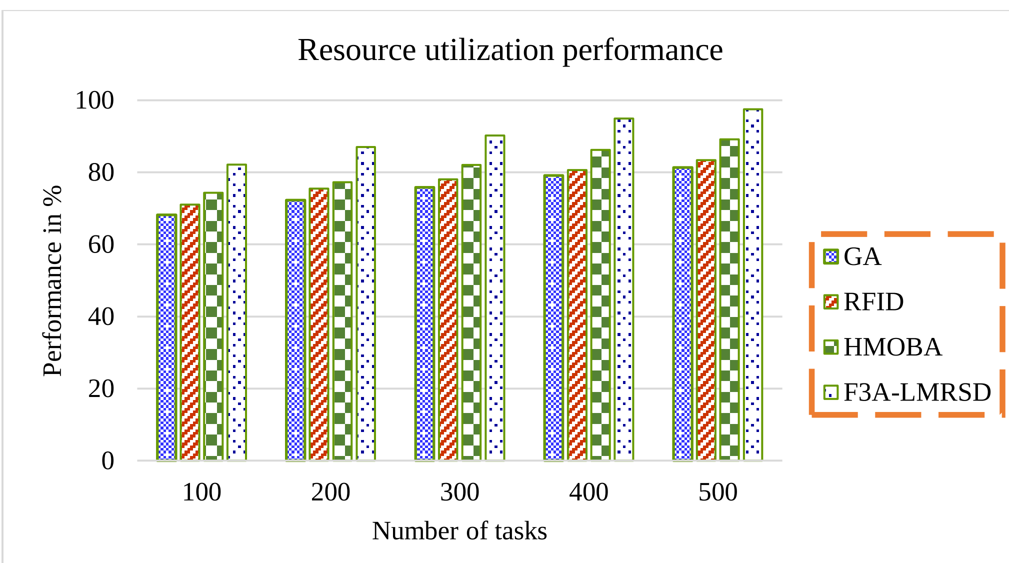

Fig.8. Analysis the Resource utilization

In comparison to other techniques, the analysis of resource utilization performance is shown in the figure. The graph clearly describes how the proposed method outshines the previous algorithm in many aspects. The proposed obtained resource utilization performance is 97.49%, while the existing methods, GA, RFID, and HMOBA, had a resource utilization performance of 81.3%, 83.34%, and 89.15%.

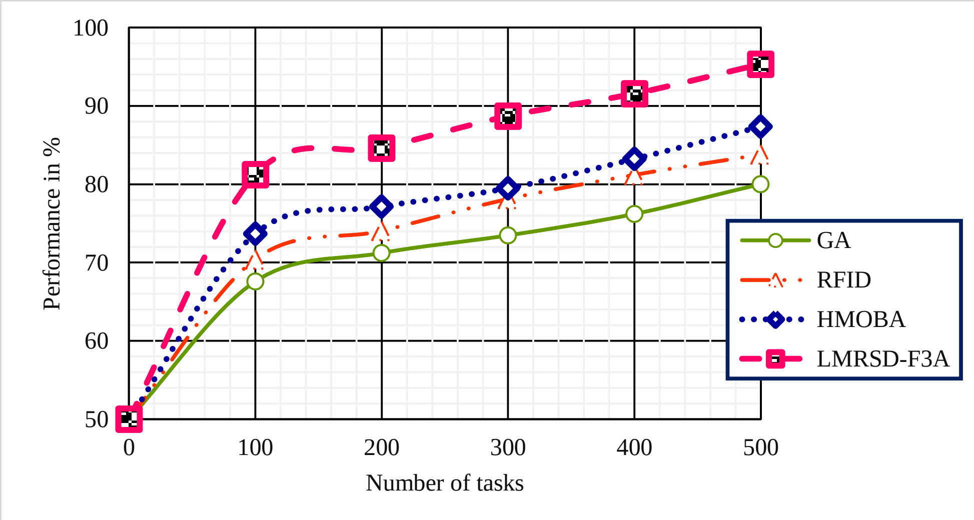

Fig. 9. Comparison of the workload balancing performance

Figure demonstrate the comparison of workload balancing performance among the proposed and other methods. The experiment output indicate that the proposed system outshines other methods in all aspects. For example, for 500 tasks, the F3A-LMRSD method achieved a 95.27% scheduling performance, while conventional methods like GA, RFID, and HMOBA achieved 79.08%, 82.16%, and 86.13%, respectively.

TABLE VII
COMPARISON OF OVERALL PERFORMANCE

Author/ Ref	Method/metrics	Scheduling performance (%)	Migration time performance (ms)	Resource utilization (%)	Workload balancing performance (%)
J. Xuan et al. [30]	GA	79.08	42.16	81.3	80.03
Z. Liu et al. [11]	RFID	82.16	33.48	83.34	83.8
B. Saemi et al. [29]	HMOBA	86.13	31.002	89.15	87.36
Proposed method	F3A-LMRSD	95.27	23.33	97.49	95.35

Table describes the comparison of overall performance for efficient task development and source distribution in the cloud environment.

V. CONCLUSION

This paper presents LMRSD approach based on the F3A for efficient task preparation and source allocation in the CE. Firstly, the proposed M2TPR algorithm divided the incoming tasks. After that, MCRA method ordered the tasks based divided tasks. Moreover, efficiently scheduled the task utilized by F3A method. Based on the scheduled task, the proposed LMRDS algorithm proficiently allotted the VMs. The proposed simulation achieved simulation metrics is scheduling performance of 95.27%, resource utilization 97.49%, workload balancing performance of 95.35% and migration time 23.33^{2x}ms. This method performs better than competing algorithms.

REFERENCES

[1]. Gawali, M.B., Shinde, S.K. Task scheduling and resource allocation in cloud computing using a heuristic approach. J Cloud Comp 7, 4 (2018).

[2]. T. Aladwani, 'Types of Task Scheduling Algorithms in Cloud Computing Environment', Scheduling Problems - New Applications and Trends. IntechOpen, Jul. 08, 2020. doi: 10.5772/intechopen.86873.

[3]. Anupama K Channappa, Nagaraja Ramaiah, Shivakumar B Rajanna "Multi-Objective Optimization Method for Task Scheduling and Resource Allocation in Cloud Environment", International Information and Engineering Technology Associations, Vol. 36, No. 2, April, 2022, pp. 223-232.

[4]. Ergu, D., Kou, G., Peng, Y. et al. The analytic hierarchy process: task scheduling and resource allocation in cloud computing environment. J Supercomput 64, 835–848 (2013). https://doi.org/10.1007/s11227-011-0625-1.

[5]. Lei Zhang and Jin-he Zhou, "Task scheduling and resource allocation algorithm in cloud computing system based on non-cooperative game," 2017 IEEE 2nd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), Chengdu, 2017, pp. 254-259, doi: 10.1109/ICCCBDA.2017.7951920.

[6]. Rajni Aron, Ajith Abraham, Resource scheduling methods for cloud computing environment: The role of meta-heuristics and artificial intelligence, Engineering Applications of Artificial Intelligence, Volume 116, 2022.

[7]. Z. Chen, J. Hu, X. Chen, J. Hu, X. Zheng and G. Min, "Computation Offloading and Task Scheduling for DNN-Based Applications in Cloud-Edge Computing, 2020.

[8]. Y. Yu and Y. Su, "Cloud Task Scheduling Algorithm Based on Three Queues and Dynamic Priority," 2019.

[9]. P. Zhang and M. Zhou, "Dynamic Cloud Task Scheduling Based on a Two-Stage Strategy, April 2018.

[10]. Y. Xiong, S. Huang, M. Wu, J. She and K. Jiang, "A Johnson's-Rule-Based Genetic Algorithm for Two-Stage-Task Scheduling Problem in Data-Centers of Cloud Computing," 2019.

[11]. B. A. Al-Maytami, P. Fan, A. Hussain, T. Baker and P. Liatsis, "A Task Scheduling Algorithm With Improved Makespan Based on Prediction of Tasks Computation Time algorithm for Cloud Computing," 2019.

[12]. L. Chen, K. Guo, G. Fan, C. Wang and S. Song, "Resource Constrained Profit Optimization Method for Task Scheduling in Edge Cloud," 2020.

[13]. H. Zhang and R. Jia, "Application of Chaotic Cat Swarm Optimization in Cloud Computing Multi Objective Task Scheduling," 2023.

[14]. S. Pang, W. Li, H. He, Z. Shan and X. Wang, "An EDA-GA Hybrid Algorithm for Multi-Objective Task Scheduling in Cloud Computing," in IEEE Access, vol. 7, pp. 146379-146389, 2019, doi: 10.1109/ACCESS.2019.2946216.

[15]. S. Pang, W. Li, H. He, Z. Shan and X. Wang, "An EDA-GA Hybrid Algorithm for Multi-Objective Task Scheduling in Cloud Computing," in IEEE Access, vol. 7, pp. 146379-146389, 2019, doi: 10.1109/ACCESS.2019.2946216.

[16]. K. Kang, D. Ding, H. Xie, Q. Yin and J. Zeng, "Adaptive DRL-Based Task Scheduling for Energy-Efficient Cloud Computing," in IEEE Transactions on Network and Service Management, vol. 19, no. 4, pp. 4948-4961, Dec. 2022, doi: 10.1109/TNSM.2021.3137926.

[17]. Z. Liu, M. Liwang, S. Hosseinalipour, H. Dai, Z. Gao and L. Huang, "RFID: Towards Low Latency and Reliable DAG Task Scheduling Over Dynamic Vehicular Clouds," in IEEE Transactions on Vehicular Technology, vol. 72, no. 9, pp. 12139-12153, Sept. 2023, doi: 10.1109/TVT.2023.3266582.

[18]. H. Yuan, J. Bi and M. Zhou, "Energy-Efficient and QoS-Optimized Adaptive Task Scheduling and Management in Clouds," in IEEE Transactions on Automation Science and Engineering, vol. 19, no. 2, pp. 1233-1244, April 2022, doi: 10.1109/TASE.2020.3042409.

[19]. L. Zhao, Y. Yang, A. Munir, A. X. Liu, Y. Li and W. Qu, "Optimizing Geo-Distributed Data Analytics with Coordinated Task Scheduling and Routing," in IEEE Transactions on Parallel and Distributed Systems, vol. 31, no. 2, pp. 279-293, 1 Feb. 2020, doi: 10.1109/TPDS.2019.2938164.

[20]. H. Yuan, J. Bi and M. Zhou, "Temporal Task Scheduling of Multiple Delay-Constrained Applications in Green Hybrid Cloud," in IEEE Transactions on Services Computing, vol. 14, no. 5, pp. 1558-1570, 1 Sept.-Oct. 2021, doi: 10.1109/TSC.2018.2878561.

[21]. S. Mousavi, S. E. Mood, A. Souri and M. M. Javidi, "Directed Search: A New Operator in NSGA-II for Task Scheduling in IoT Based on Cloud-Fog Computing," in IEEE Transactions on Cloud Computing, vol. 11, no. 2, pp. 2144-2157, 1 April-June 2023, doi: 10.1109/TCC.2022.3188926.

[22]. P. Banerjee et al., "MTD-DHJS: Makespan-Optimized Task Scheduling Algorithm for Cloud Computing With Dynamic Computational Time Prediction," in IEEE Access, vol. 11, pp. 105578-105618, 2023, doi: 10.1109/ACCESS.2023.3318553.

[23]. H. Mahmoud, M. Thabet, M. H. Khafagy and F. A. Omara, "Multiobjective Task Scheduling in Cloud Environment Using Decision Tree Algorithm," in IEEE Access, vol. 10, pp. 36140-36151, 2022, doi: 10.1109/ACCESS.2022.3163273.

[24]. Y. Alahmad, T. Daradkeh and A. Agarwal, "Proactive Failure-Aware Task Scheduling Framework for Cloud Computing," in IEEE Access, vol. 9, pp. 106152-106168, 2021, doi: 10.1109/ACCESS.2021.3101147.

[25]. X. Chen et al., "A WOA-Based Optimization Approach for Task Scheduling in Cloud Computing Systems," in IEEE Systems Journal, vol. 14, no. 3, pp. 3117-3128, Sept. 2020, doi: 10.1109/JSYST.2019.2960088.

[26]. S. Lipsa, R. K. Dash, N. Ivković and K. Cengiz, "Task Scheduling in Cloud Computing: A Priority-Based Heuristic Approach," in IEEE Access, vol. 11, pp. 27111-27126, 2023, doi: 10.1109/ACCESS.2023.3255781.

[27]. S. Geng, D. Wu, P. Wang and X. Cai, "Many-Objective Cloud Task Scheduling," in IEEE Access, vol. 8, pp. 79079-79088, 2020, doi: 10.1109/ACCESS.2020.2990500.

[28]. D. Alsadie, "TSMGWO: Optimizing Task Schedule Using Multi-Objectives Grey Wolf Optimizer for Cloud Data Centers," in IEEE Access, vol. 9, pp. 37707-37725, 2021, doi: 10.1109/ACCESS.2021.3063723.

[29]. J. Sun et al., "Multiobjective Task Scheduling for Energy-Efficient Cloud Implementation of Hyperspectral Image Classification," in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 14, pp. 587-600, 2021, doi: 10.1109/JSTARS.2020.3036896.

[30]. E. Cao et al., "Energy and Reliability-Aware Task Scheduling for Cost Optimization of DVFS-Enabled Cloud Workflows," in IEEE Transactions on Cloud Computing, vol. 11, no. 2, pp. 2127-2143, 1 April-June 2023, doi: 10.1109/TCC.2022.3188672.

[31]. L. Liu, Q. Fan and R. Buyya, "A Deadline-Constrained Multi-Objective Task Scheduling Algorithm in Mobile Cloud Environments," in IEEE Access, vol. 6, pp. 52982-52996, 2018, doi: 10.1109/ACCESS.2018.2870915.

[32]. F. A. Saif, R. Latip, Z. M. Hanapi and K. Shafinah, "Multi-Objective Grey Wolf Optimizer Algorithm for Task Scheduling in Cloud-Fog Computing," in IEEE Access, vol. 11, pp. 20635-20646, 2023, doi: 10.1109/ACCESS.2023.3241240.

[33]. K. Li, L. Jia and X. Shi, "Research on Cloud Computing Task Scheduling Based on PSOMC," in Journal of Web Engineering, vol. 21, no. 6, pp. 1749-1766, September 2022, doi: 10.13052/jwe1540-9589.2161.

[34]. C. Sun, X. Wen, Z. Lu, W. Jing and M. Zorzi, "Eco-Friendly Powering and Delay-Aware Task Scheduling in Geo-Distributed Edge-Cloud System: A Two-Timescale Framework," in IEEE Access, vol. 8, pp. 96468-96486, 2020, doi: 10.1109/ACCESS.2020.2995806.

[35]. B. Saemi, A. A. R. Hosseinabadi, A. Khodadadi, S. Mirkamali and A. Abraham, "Solving Task Scheduling Problem in Mobile Cloud Computing Using the Hybrid Multi-Objective Harris Hawks Optimization Algorithm," in IEEE Access, vol. 11, pp. 125033-125054, 2023, doi: 10.1109/ACCESS.2023.3329069.

[36]. J. Xuan, L. Chengyang and X. Jiang, "Research on Multi-objective Fuzzy Flexible Job-Shop Scheduling Based on Cloud Computinger," 2020 IEEE 8th International Conference on Computer Science and Network Technology (ICCSNT), Dalian, China, 2020, pp. 7-10, doi: 10.1109/ICCSNT50940.2020.9305019.

