International Journal of Engineering Research and Science & Technology

www.ijerst.org

ISSN : 2319-5991

Vol. 21 No. 3 (1) 2025

ijerst.editor@gmail.com

editor@ijerst.com

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 21, No. 3(1), 2025

Research Paper

A STRUCTURED TWO-PHASE MODEL FOR EFFICIENT IOT-FOG-CLOUD RESOURCE ALLOCATION

Sana Firdous[1], Dr. Mohd Umar Farooq[2]

[1]PG Scholar, Department of CS, Shadan Women's College of Engineering and Technology, Hyderabad, sanafirdous6323@gmail.com

[2]Professor, Department of CSE, Shadan Women's College of Engineering and Technology umarfarooq.mohd@gmail.com

ABSTRACT:

The amount of data produced by intelligent devices has grown significantly with the introduction of the Internet of Things (IoT) concept and the rapid advancement of technology. To analyze and store the created data, cloud computing offers limitless processing and storage capacity. Nevertheless, the cloud computing paradigm is linked to a lack of geographical awareness, excessive energy consumption, and significant transmission delay. However, the data produced by the intelligent devices must be analysed instantly because it is delay-sensitive. Cloud computing is therefore unsuitable for handling this delay-sensitive data. The fog paradigm, which enables data processing close to IoT devices, was proposed to mitigate the problems with the cloud paradigm. The fog paradigm's constraints, which render it unsuited for processing massive amounts of data, are one of its common characteristics. The fog paradigm must work with the cloud paradigm to accomplish a shared objective in order to guarantee the efficient completion of operations involving delay-sensitive applications and the massive amount of data created. In order to effectively and efficiently employ the fog and cloud resources for carrying out operations that are sensitive to delays and the massive amount of data produced by end users, an efficient resource allocation system is put forth in this study. There are two steps involved in assigning resources to assignments. Prior to being assigned to appropriate resources in the layers of their respective classes, the tasks in the arrival queue are first categorized according to the task guarantee ratio on the cloud and fog layers. Second, in order to categorize freshly arrived tasks and assign appropriate resources to the tasks for execution in the layers of their respective classes, we apply Bayes' classifier using prior allocation history data. The system's execution time and latency are decreased by creating an ideal resource allocation in the fog and cloud layers using a Crayfish Optimization Algorithm (COA). The iFogSim simulator toolkit is used to construct the suggested method, and the execution results show greater promise than the state-of-the-art techniques.

Received: 23-07-2025 Accepted: 28-08-2025 Published: 05-9-2025

1. INTRODUCTION

The exponential growth in information technology has resulted in the development of a large number of intelligent devices that can be connected together to achieve a common goal. The development of these intelligent devices further necessitates the need for several computation paradigms, like cloud, fog, and edge, among others, and a stack of communication protocols that govern their interactions. The amount of data being injected into the Internet has significantly increased due to its expanding network. One of the emerging technologies that shows promise is cloud computing, which offers limitless capacity for processing and storing the massive volumes of data produced. It is the leading technology that draws in commercial applications and services due to its scalability, flexibility, affordability, and ease of use [2], [3]. Distributed cloud networking and resource virtualization give cloud computing its competitive advantages [4]. Users can access the cloud's limitless computing, storage, apps, and other resources over the internet, which dramatically reduces on the expense and strain of keeping resources apart [5].Notwithstanding the benefits of the cloud, the paradigm is linked to significant transmission delay, which is caused by the separation between cloud data centers and consumers. However, cloud computing is not appropriate for processing delay-sensitive data, which is produced by IoT devices instead.

In 2012, Cisco unveiled fog computing, a new computing paradigm designed to satisfy the Quality of Service (QoS) demands of these delay-sensitive tasks [7].In order to provide cloud-like services close to edge users, fog technology created a highly virtualized platform that lies between cloud datacentres and edge users [8, 9, 10, 11]. By using a variety of data sources either locally within the fog cluster or by sending them to cloud datacentres, fog computing processes the data produced by the intelligent devices [12]. The fog's capacity to digest the data produced close to intelligent equipment is responsible for its quicker reaction time and higher quality of service [7].The fog paradigm's constraints, which render it unsuited for processing massive amounts of data, are one of its common characteristics. The fog paradigm must work with the cloud paradigm to guarantee the efficient completion

of latency-sensitive application tasks and the massive amount of data created.

Due to the cloud's high latency, the computation and storage limitations of fog resources, and the significant rise in data generated by IoT devices, an effective resource allocation scheme is required that divides tasks among fog and cloud resources while accounting for the tasks' quality of service (QoS) requirements. To take advantage of the benefits provided by the fog and cloud paradigms, a number of resource allocation strategies have been put forth based on the literature. For instance, while allocating the available fog resources to jobs, the authors of [13], [14], and [15] consider the resource's response time, processing speed, and bandwidth. Unallocated tasks are sent to the cloud for processing when there are no idle resources in the fog layer. Due to a scarcity of resources in the fog, these methods can, nevertheless, assign delay-sensitive tasks that are queued behind delay-insensitive tasks to the cloud and assign delay-insensitive tasks to the fog according to their arrival time.

To effectively and efficiently employ the fog and cloud resources for carrying out delay-sensitive operations and the massive amount of data produced by end users, an efficient resource allocation system is put forth in this study. There are two steps involved in allocating resources to tasks. First, the task guarantee ratio on the cloud and fog layers is used to categorize the jobs in the arrival queue. A Crayfish Optimization According to their class layer, the classified jobs are given the appropriate resources using a Crayfish Optimization Algorithm (COA). Second, we classify recently arriving tasks and pair them with appropriate resources for execution in the layer of their respective classes by using Bayes' classifier to historical allocation history data. This method gives the computation-intensive and data-intensive tasks access to the cloud's limitless processing and storage capacity, while the delay-sensitive chores benefit from the fog's quicker response and higher QoS. By cutting down on resource search time in both the fog and cloud levels, the suggested method can greatly cut down on scheduling time and enhance overall system performance.

PARTICIPATION

The following are the contributions made by this paper: To guarantee the best workload allocation between the fog and cloud resources, we first present a classifier based on the task guarantee ratio and the Bayes theorem. After a thorough search for the best layers and resources to carry out tasks, the Bayes classification was developed to lower system overhead. Second, the resource allocation problem is formulated as a multi-objective optimization model that takes task execution time and latency into account. To produce the best resource allocation that lowers the overall system execution time and latency, we present the crayfish optimization algorithm, which draws inspiration from the foraging behaviour of crayfish.

ESSENTIAL WORKS

Several scheduling strategies have been put out to address how user tasks are impacted by the lengthy transmission delay connected to cloud computing and the resource limitations connected to fog. For instance, in order to improve the performance of the whale optimization algorithm (WOA) and handle scheduling issues in fog computing, the authors of [16] presented an opposition-based chaotic whale optimization algorithm (OppoCWOA). The authors take into account the time and energy usage in a fog computing environment when formulating the job scheduling problem as a linear programming problem. The suggested method shows promise in terms of energy and time usage. But the suggested system didn't Think about using the cloud to schedule tasks, as this could cause the fog layer to become overloaded.

A task scheduler based on linear programming that distributes resources to tasks on the cloud or fog layer was proposed by the authors in [17]. The tasks' service class determines how resources are allocated. The processing components to be used for the task are chosen by a service class.Real-time, mission-critical, interactive, conversational, streaming, CPU-bound, and best-effort are among the service classifications taken into consideration. After the suggested strategy was put into practice, the numerical results beat a number of cutting-edge techniques. However, since the job classes are known ahead of time, direct class-to-resource mapping can be used to allocate layer resources.

OBJECTIVE

The authors formulate the task scheduling problem as a linear programming problem, taking into consideration the time and energy consumption in a fog computing environment. The service classes considered include real-time, mission critical, interactive, CPU-bound, and best-effort.

PROBLEM STATEMENT

The rise of the IoT has led to an exponential increase in data generated by intelligent devices, creating challenges in processing and storage. While cloud computing offers extensive processing and storage capabilities, it suffers from high transmission latency, high energy consumption, and a lack of location awareness, making it unsuitable for delay-sensitive data processing. In contrast, the fog computing paradigm addresses these issues by enabling data processing near IoT devices, though it has limitations in handling large data volumes. To overcome these challenges, this paper proposes an efficient resource allocation framework that combines fog and cloud resources to process delay-sensitive tasks and large data volumes effectively.

EXISTING SYSTEM

The amount of data produced by intelligent devices has grown significantly with the introduction of the Internet of Things (IoT) concept and the rapid advancement of technology. To analyze and store the created data, cloud computing offers limitless processing and storage capacity. Nevertheless, the cloud computing paradigm is linked to a lack of geographical awareness, excessive energy consumption, and significant transmission delay. However, the data produced by the intelligent devices must be analysed instantly because it is delay-sensitive. Cloud computing is therefore unsuitable for carrying out this delay-sensitive data. The fog paradigm, which enables data processing close to IoT devices, was proposed to mitigate the problems with the cloud paradigm. The fog paradigm's constraints, which render it unsuited for processing massive amounts of data, are one of its common characteristics. The fog paradigm must work with the cloud paradigm to accomplish a shared objective in order to guarantee the efficient completion of operations involving delay-sensitive applications and the massive amount of data created.

DISADVANTAGE

- The flexibility of edge computing makes the environment more complex.
- The current edge computing architecture has the problems.
- The edge node carries too many task requests

PROPOSED SYSTEM

In order to effectively and efficiently employ the fog and cloud resources for carrying out operations that are sensitive to delays and the massive amount of data produced by end users, an efficient resource allocation system is put forth in this study. There are two steps involved in assigning resources to assignments. Prior to being assigned to appropriate resources in the layers of their respective classes, the tasks in the arrival queue are first categorized according to the task guarantee ratio on the cloud and fog layers. Second, in order to categorize freshly arrived tasks and assign appropriate resources to the tasks for execution in the layers of their respective classes, we apply Bayes' classifier using prior allocation history data. The system's execution time and latency are decreased by creating an ideal resource allocation in the fog and cloud layers using a Crayfish Optimization Algorithm (COA).

ADVANTAGES

- High bandwidth and low latency
- Significantly improve the real-time experience and satisfaction of users.
- Edge computing nodes are scattered.

2.RELATED WORKS

The introduction of cloud computing has brought about significant developments in information technology. Users can benefit from the multitude of cloud technology services only by connecting to the internet. In cloud computing, load balancing is the fundamental issue that has challenged experts in this research area. Load balancing helps increase user satisfaction and enhance systems' productivity through efficient and fair work assignments between computing resources. Besides, maintaining a load balancing among resources would be difficult because the resources are usually distributed in a heterogeneous way. Many load-balancing methods try to solve this problem by the metaheuristics algorithm, and each of them attempted to enhance the operation and efficiency of systems. In this paper, grey wolf optimization (GWO) algorithm has been used based on the resource reliability capability to maintain proper load balancing. The results of simulation in CloudSim showed that the costs and response time in the proposed method are less than the other methods, and the obtained solutions are ideal.**[1]**

Cloud manufacturing (CMfg) has received increasingly attention from both academia and industry. Cloud service composition is a critical technique in CMfg that connects different available manufacturing cloud services (MCSs) to generate a composite manufacturing cloud service (CMCS) to satisfy users' requirements. Many available MCSs with the same or similar functionality but different QoS attributes are deployed in the CMfg platform. In this approach, the uniform mutation is applied to perform the global search to preserve the diversification of the population, and a modified whale optimization algorithm is designed to perform the local search. The performance of the new approach is verified on various benchmark functions and different scales of QoS-aware cloud service composition problems. The experimental results demonstrate that the proposed MWOA has superior performance over the other methods.**[2]**

Information and technology have witnessed significant improvement with the introduction of Internet of things (IoT) applications, and most of the IoT applications are dependent on the cloud. Scheduling tasks based on MIPs size and prioritizing the tasks with smaller MIPs size first make critical tasks with larger MIPs wait, which ultimately increases the delay and may result in some serious problems. This paper proposed a scheduler "Critical task First Scheduler" (CTFS), which schedules tasks depending on the nature of the requests, which are classified as either critical or noncritical. The environment was kept the same for all the approaches that are implemented to demonstrate the effectiveness of the proposed approach. The results of the proposed approach were compared with First Come First Served (FCFS), Shortest Job First (SJF), and cloud-only approaches to demonstrate the effectiveness of

the proposed approach in terms of latency, energy consumption, and network utilization. Simulation results show that the proposed CTFS approach outperformed the compared techniques for all three comparison parameters.**[3]**

This paper proposes a meta heuristic optimization algorithm, called Crayfish Optimization Algorithm (COA), which simulates crayfish's summer resort behavior, competition behavior and foraging behavior. The three behaviors are divided into three different stages to balance the exploration and exploitation of algorithm. The three stages are summer resort stage, competition stage and foraging stage. The summer resort stage represents the exploration stage of the COA. The competition stage and foraging stage represent the exploitation stage of the COA. To verify the optimization effect of COA, in the experimental part, 23 standard benchmark functions and CEC2014 benchmark functions are used to test, and 9 algorithms are selected for comparative experiments. The experimental results show that COA can balance the exploration and exploitation, and achieve good optimization effect.**[4]**

Fog computing extends the facility of cloud computing from the center to edge networks. Although fog computing has the advantages of location awareness and low latency, the rising requirements of ubiquitous connectivity and ultra-low latency challenge real-time traffic management for smart cities. As an integration of fog computing and vehicular networks, vehicular fog computing (VFC) is promising to achieve real-time and location-aware network responses. Furthermore, the VFC-enabled offloading scheme is formulated as an optimization problem by leveraging moving and parked vehicles as fog nodes. A real-world taxi-trajectory-based performance analysis validates our model. Finally, some research challenges and open issues toward VFC-enabled traffic management are summarized and highlighted.**[5]**

The amount of data produced by intelligent devices has grown significantly with the introduction of the Internet of Things (IoT) concept and the rapid advancement of technology. To analyze and store the created data, cloud computing offers limitless processing and storage capacity. Nevertheless, the cloud computing paradigm is linked to a lack of geographical awareness, excessive energy consumption, and significant transmission delay. However, the data produced by the intelligent devices must be analysed instantly because it is delay-sensitive. The system's execution time and latency are decreased by creating an ideal resource allocation in the fog and cloud layers using a Crayfish Optimization Algorithm (COA). The iFogSim simulator toolkit is used to construct the suggested method, and the execution results show greater promise than the state-of-the-art techniques.**[6]**

"An Efficient Population-Based Multi-Objective Task Scheduling Approach in Fog Computing Systems", presents a novel task scheduling algorithm tailored for fog computing environments. The proposed approach addresses the challenges of resource constraints, latency sensitivity, and energy consumption by formulating the scheduling process as a multi-objective optimization problem. By leveraging a population-based metaheuristic algorithm, the method aims to optimize task execution time, energy usage, and overall system performance. The authors conduct extensive simulations to evaluate the effectiveness of their approach and demonstrate its superiority over existing scheduling methods in terms of improving resource utilization, minimizing latency, and achieving better energy efficiency in fog computing systems.**[7]**

This research paper presents a novel approach to enhancing Quality-of-Service (QoS) in fog computing environments by proposing an efficient resource allocation strategy. The authors address key challenges such as latency, bandwidth limitations, and dynamic workloads by developing a task-aware resource management framework. The model considers task characteristics and available fog resources to optimize task distribution and minimize service delays. Through simulations, the proposed method demonstrates improved performance in terms of reduced task execution time, better resource utilization, and increased reliability, making it a valuable contribution to real-time and delay-sensitive applications in IoT and fog computing ecosystems.**[8]**

The paper titled *"iFogSim: A Toolkit for Modeling and Simulation of Resource Management Techniques in the Internet of Things, Edge and Fog Computing Environments"* presents a simulation framework designed to evaluate performance and efficiency of resource management strategies in fog and edge computing systems. The authors introduce iFogSim as a flexible and extensible toolkit that helps researchers and developers analyze latency, energy consumption, and network usage in distributed computing setups involving IoT devices. Overall, the toolkit aims to facilitate the development and optimization of IoT applications by enabling simulation of real-world deployments and resource coordination strategies.**[9]**

The research by R. Mehta, J. Sahni, and K. Khanna focuses on optimizing task scheduling to enhance the responsiveness of latency-sensitive applications in fog-cloud integrated environments. The paper introduces a scheduling framework that reduces response time by strategically assigning tasks across fog and cloud resources based on application urgency and resource availability. By analyzing the characteristics of tasks and the dynamic nature of fog and cloud layers, the authors develop a method that minimizes latency while maintaining service quality. The proposed approach is evaluated through simulations, showing notable improvements in response time compared to existing techniques,

making it well-suited for real-time applications like IoT and multimedia services.**[10]**

3. METHODOLOGIES

1. User Interface Design

In this module we design the windows for the project. These windows are used for secure login for all users. To connect with server user must give their username and password then only they can able to connect the server. If the user already exits directly can login into the server else user must register their details such as username, password and Email id, into the server. Server will create the account for the entire user to maintain upload and download rate. Name will be set as user id. Logging in is usually used to enter a specific page.

2. Admin

This is the first module of this project. In this module admin can cloud a user details. The admin can have all users' data.

3. User

This is the second module of this project. In this module. user can register after register they need to be a login takes permission from cloud. The user can see all my files in the virtual machine data stores. The user can have a backup data. The user can also have a view a data. The users can have a recover a data from a deleted a data.

4. Cloud

This is the third module of this project. In this module. Cloud can register and then login. Cloud can have a user request. Cloud only allows a permission for the users. Users cannot login directly takes a permission after approve then it was login. The cloud has stored all the user details.

5.Server

This is the first module of this project. In this module admin can cloud a user details. The admin can have all users' data.

4. ALGORITHM

A. EXISTING ALGORITHM

- **BAYES CLASSIFIER**

The following are the contributions made by this paper: To guarantee the best possible workload allocation between the fog and cloud resources, we first introduce a classifier based on the task guarantee ratio and the Bayes theorem. In order to minimize system overhead caused by a thorough search for the right layers and resources for task execution, the Bayes classification was implemented. Second, the resource allocation problem is formulated as a multi-objective optimization model that takes task execution time and latency into account. To generate optimal resource allocation that minimizes the overall system execution time and latency, we introduce the crayfish optimization algorithm, which draws inspiration from the foraging behaviour of crayfish. The resource allocator receives the classified tasks and chooses the optimal resource for each task within its class layer. In order to classify a new job based on the allocation history data, the classifier uses Bayesian classification.

B. PROPOSED ALGORITHM

- **A NOVEL CRAYFISH OPTIMISATION ALGORITHM (COA)**

An ideal resource allocation in the fog and cloud layers is produced using a Crayfish Optimization Algorithm (COA), which lowers the system's execution time and delay. When the suggested approach is used, the execution outcomes show greater promise than those of the state-of-the-art techniques. The classified jobs are given the appropriate resources according to their class layer using a Crayfish Optimization Algorithm (COA). To classify newly arrived tasks and pair them with appropriate resources for execution in the layer of their respective classes, we secondly apply to data from prior allocation histories. In COA, arbitrary constants that reflect the temperature of the several areas that each individual crayfish resides in govern the exploitation and exploration stages.

- **Virtual machine algorithm**

Allocating virtual machines to physical equipment can be thought of as a cloud provider optimization problem. The distribution of virtual machines (VMs) need to optimize the load while maintaining the caliber of service supply. The following is a statement of the optimization problem:

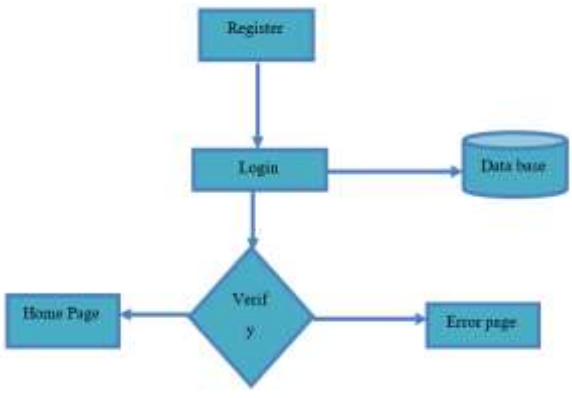
Let VMs be represented by V1, V2,... and PMs in a data provided by P1, P2, P3. This means that a virtual machine Vi is only linked to a physical machine Pi if Pi has at least as many of these as the virtual machine Vi. The pseudo code is used to show the algorithm for the aforementioned assignment problem.

Input: a set of virtual machines Vi and a set of physical machines Pj

Output: a set of (Vi, Pj), i.e. VMs to a PM allocation pair

For each virtual machine Vi that is not assigned to a physical machine Pi.

5. DATAFLOW DIAGRAM



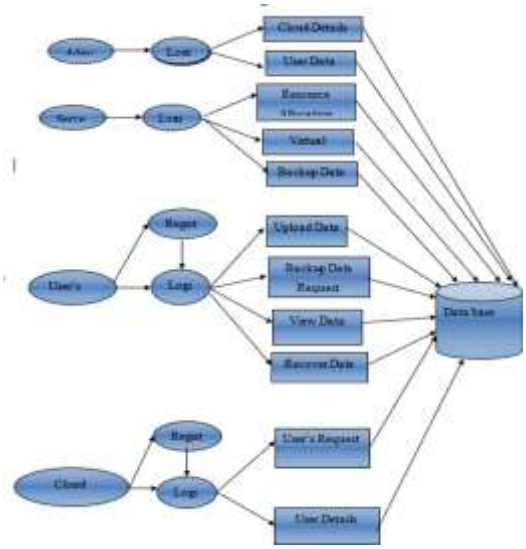


Fig 5: Dataflow Diagram

6. SYSTEM ARCHITECTURE

Fig 6: System Architecture

This is a diagram of a cloud data management system in which various actors engage a central database. Admin oversees user data and cloud specifics, the Server is responsible for resource allocation, backups, and virtual machines, the User is able to sign up, upload data, initiate backups, view, and retrieve data, and the Cloud Provider oversees requests and user details. All these activities are interlinked via the Database, which is the central point of information storage and retrieval

7. RESULTS

The suggested fog–cloud resource allocation system enhances performance for delay-constrained IoT applications with faster execution time and reduced end-to-end latency. Utilizing task grouping, Bayes' classifier, and the Crayfish Optimization Algorithm (COA), it facilitates effective load balancing, optimal resource usage, and quick response over conventional cloud-only schemes.

Fig :7: Accuracy

8. FUTURE IMPROVEMENT

Additionally, by using historical data to anticipate the layer and resource of newly arrived tasks, the system overhead caused by the optimal resource search time can be decreased. The iFogSim simulator was used to implement the suggested approach in addition to additional approaches including cloud-only, fog-only, and random methods. The suggested strategy performed better than the alternative approaches in terms of task failure, latency, and execution time, according to the simulation findings. It is clear from the performance shown by the suggested method that not all tasks can be completed with fog alone without the cloud, and vice versa.

9. CONCLUSION

An effective resource allocation strategy that guarantees the proper distribution and execution of tasks across the fog and cloud layers is required due to the significant transmission latency experienced by delay-sensitive tasks when executed on the cloud and the lengthy execution time experienced by computation-intensive and data-intensive tasks when executed on the fog layer. In order to effectively and efficiently employ the fog and cloud resources for carrying out operations that are sensitive to delays and the massive amount of data produced by end users, an efficient resource allocation system is put forth in this study. There are two phases to the distribution of resources among tasks. First, using the COA method, the tasks in the arrival queue are categorized according to the task guarantee ratio on the cloud and fog layers, and appropriate resources are assigned in the layers of their respective classes. Second, we use Bayes' classifier on historical allocation data to categorize newly arrived tasks and assign them to the appropriate class-specific layer resources. In particular, when the system has a lot of tasks to do, the use of classifiers to categorize following jobs can minimize the inaccuracy in resource allocation.

10.REFERENCES

[1] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the Internet of Things: A survey," IEEE Commun. Surveys Tuts., vol. 16, no. 1, pp. 414–454, 1st Quart., 2014, doi: 10.1109/SURV.2013.042313.00197.

[2] S. Sefati, M. Mousavinasab, and R. Zareh Farkhady, "Load balancing in cloud computing environment using the grey wolf optimization

algorithm based on the reliability: Performance evaluation," J. Supercomput., vol. 78, no. 1, pp. 18–42, May 2021, doi: 10.1007/s11227-021-03810-8.
[3] A. Najafizadeh, A. Salajegheh, A. M. Rahmani, and A. Sahafi, "Multiobjective task scheduling in cloud-fog computing using goal programming approach," Cluster Comput., vol. 25, no. 1, pp. 141–165, Aug. 2021, doi: 10.1007/s10586-021-03371-8.
[4] H. Jin, S. Lv, Z. Yang, and Y. Liu, "Eagle strategy using uniform mutation and modified whale optimization algorithm for QoS-aware cloud service composition," Appl. Soft Comput., vol. 114, Jan. 2022, Art. no. 108053, doi: 10.1016/j.asoc.2021.108053.
[5] Z. Tong, X. Deng, H. Chen, and J. Mei, "DDMTS: A novel dynamic load balancing scheduling scheme under SLA constraints in cloud computing," J. Parallel Distrib. Comput., vol. 149, pp. 138–148, Mar. 2021, doi: 10.1016/j.jpdc.2020.11.007.
[6] Z. Ning, J. Huang, and X. Wang, "Vehicular fog computing: Enabling real-time traffic management for smart cities," IEEE Wireless Commun., vol. 26, no. 1, pp. 87–93, Feb. 2019, doi: 10.1109/MWC.2019.1700441.
[7] H. Atlam, R. Walters, and G. Wills, "Fog computing and the Internet of Things: A review," Big Data Cognit. Comput., vol. 2, no. 2, p. 10, Apr. 2018, doi: 10.3390/bdcc2020010.
[8] F. Bonomi, R. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in Proc. MCCWorkshop Mobile cloud Comput., pp. 13–15, Aug. 2012, doi: 10.1145/2342509.2342513.
[9] C. V. N. Angeline and R. Lavanya, "Fog computing and its role in the Internet of Things," in Advancing Consumer-centric Fog Computing Architecture, K. Munir, Ed. Harshey, PA, USA: IGI Global, 2019, pp. 63–71.
[10] M. Saad, "Fog computing and its role in the Internet of Things: Concept, security and privacy issues," Int. J. Comput. Appl., vol. 180, no. 32, pp. 7–9, Apr. 2018, doi: 10.5120/ijca2018916829.
[11] B. Negash, A. M. Rahmani, P. Liljeberg, and A. Jantsch, "Fog computing fundamentals in the Internet-of-Things," in Fog Computing in the Internet of Things: Intelligence at the Edge. Cham, Switzerland: Springer, 2017, pp. 3–13, doi: 10.1007/978-3-319-57639-8_1.
[12] M. Al-khafajiy, T. Baker, H. Al-Libawy, A. Waraich, C. Chalmers, and O. Alfandi, "Fog computing framework for Internet of Things applications," in Proc. 11th Int. Conf. Develop. eSystems Eng. (DeSE), Sep. 2018, pp. 71–77, doi: 10.1109/DeSE.2018.00017.
[13] R. K. Naha, S. Garg, A. Chan, and S. K. Battula, "Deadline-based dynamic resource allocation and provisioning algorithms in fog-cloud environment," Future Gener. Comput. Syst., vol. 104, pp. 131–141, Mar. 2020, doi: 10.1016/j.future.2019.10.018.
[14] S. K. Mani and I. Meenakshisundaram, "Improving quality-of-service in fog computing through efficient resource allocation," Comput. Intell., vol. 36, no. 4, pp. 1527–1547, Feb. 2020, doi: 10.1111/coin.12285.
[15] H. Rafique, M. A. Shah, S. U. Islam, T. Maqsood, S. Khan, and C. Maple, "A novel bio-inspired hybrid algorithm (NBIHA) for efficient resource management in fog computing," IEEE Access, vol. 7, pp. 115760–115773, 2019, doi: 10.1109/ACCESS.2019.2924958.
[16] Z. Movahedi, B. Defude, and A. M. Hosseininia, "An efficient populationbased multi-objective task scheduling approach in fog computing systems," J. Cloud Comput., vol. 10, no. 1, pp. 1–31, Oct. 2021, doi: 10.1186/s13677-021-00264-4.
[17] J. C. Guevara and N. L. S. da Fonseca, "Task scheduling in cloud-fog computing systems," Peer-Peer Netw. Appl., vol. 14, no. 2, pp. 962–977, Jan. 2021, Doi: 10.1007/s12083-020-01051-9.
[18] R. Mehta, J. Sahni, and K. Khanna, "Task scheduling for improved response time of latency sensitive applications in fog integrated cloud environment," Multimedia Tools Appl., vol. 82, no. 21, pp. 32305–32328, Mar. 2023, doi: 10.1007/s11042-023-14565-0.
[19] A. Khan, A. Abbas, H. A. Khattak, F. Rehman, I. U. Din, and S. Ali, "Effective task scheduling in critical fog applications," Sci. Program., vol. 2022, pp. 1–15, Mar. 2022, Doi: 10.1155/2022/9208066.
[20] N. Potu, C. Jatoth, and P. Parvataneni, "Optimizing resource scheduling based on extended particle swarm optimization in fog computing environments," Concurrency Comput., Pract. Exper., vol. 33, no. 23, pp. 1–13, Jan. 2021, doi: 10.1002/cpe.6163.
[21] A. Abouaomar, S. Cherkaoui, A.Kobbane, and O. A. Dambri, "Aresources representation for resource allocation in fog computing networks," in Proc. IEEE Global Commun. Conf. (GLOBECOM), Dec. 2019, pp. 1–6, doi: 10.1109/GLOBECOM38437.2019.9014146.
[22] H. Mohammady Talvar, H. Haj Seyyed Javadi, H. Navidi, and A. Rezakhani, "A new resource allocation method in fog computing via non-cooperative game theory," J. Intell. Fuzzy Syst., vol. 41, no. 2, pp. 3921–3932, Sep. 2021, doi: 10.3233/jifs-202122.
[23] M. Iyapparaja, N. K. Alshammari, M. S. Kumar, S. S. R. Krishnan, and C. L. Chowdhary, "Efficient resource allocation in fog computing using QTCS model," Comput., Mater. Continua, vol. 70, no. 2, pp. 2225–2239, 2022, doi: 10.32604/cmc.2022.015707.
[24] T. Tuithung, R. Kumar, and B. Sarma, "Optimised fuzzy clustering-based resource scheduling and dynamic load balancing algorithm for fog computing environment," Int. J. Comput. Sci. Eng., vol. 24, no. 4, p. 343, 2021, doi: 10.1504/ijcse.2021.10039964.
[25] H. Jia, H. Rao, C. Wen, and S. Mirjalili, "Crayfish optimization algorithm," Artif. Intell. Rev.,

vol. 56, no. S2, pp. 1919–1979, Sep. 2023, doi: 10.1007/s10462-023-10567-4.
[26] H. Gupta, A. Vahid Dastjerdi, S. K. Ghosh, and R. Buyya, "IFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, edge and fog computing environments," Softw., Pract. Exper., vol. 47, no. 9, pp. 1275–1296, Jun. 2017, doi: 10.1002/spe.2509