International Journal of Engineering Trends and Technology
ISSN: 2231–5381 / https://doi.org/10.14445/22315381/IJETT-V73I2P101

Volume 73 Issue 2, 1-8, February 2025
© 2025 Seventh Sense Research Group®

Original Article

Quality of Service-Driven Optimized Resource Provisioning in Fog Computing

Vadde Usha[1], T. K. Rama Krishna Rao[2]

[1,2]Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Andhra Pradesh, India.

[1]Corresponding Author : vuat5678@gmail.com

Received: 23 October 2024 Revised: 04 January 2025 Accepted: 21 January 2025 Published: 21 February 2025

Abstract *- Fog computing extends cloud services at the network edge and has become a feasible solution for delay-sensitive IoT applications. In order to prevent uneven load distribution and to ensure the quality of service, resource provisioning is an essential aspect of the fog network. As fog nodes are dynamic and heterogeneous with varying resource capabilities, effective resource provisioning mechanisms are required for the potential use of the fog environment. Since IoT applications have different QoS requirements, it is necessary to develop efficient resource provisioning techniques considering the deadlines of applications. Numerous approaches to fog computing resource provisioning exist in the literature; nevertheless, there is still scope for improvement in the performance. In this work, we propose QoS-driven resource provisioning using an enhanced Coati Optimization Algorithm (eCOA). Our proposed model aims to minimize the delay, energy, and execution cost of IoT applications while focusing on QoS application requirements. The results have shown that the proposed algorithm reduces, on average, 21% of delay,25% of energy and 24% of cost compared to other meta-heuristic algorithms.*

Keywords *- Enhanced coati optimization, Fog computing, Optimization, Quality of service, Resource provision.*

1. Introduction

The Internet of Things (IoT) paradigm is becoming more popular, resulting in a substantial rise in the number of connected smart devices and IoT applications. Cloud computing was the first workable solution for handling and storing the data generated by smart devices. Severe network bottlenecks and latency are caused when large volumes of data are sent from connected devices to the cloud [1]. In response to these problems, the fog computing paradigm was developed, whereby cloud-like services are offered at the network's edge through the use of devices with networking, storage, and compute capabilities located along the path connecting [2, 3] Fog computing often offers higher Quality of Service (QoS) concerning latency, bandwidth, reaction time, and usage of energy [4, 5]. By placing processing power between the data source and the distant cloud, fog nodes, a decentralised architecture, analyse data from the Internet of Things devices in real time [6, 7].

The successful and efficient operation of fog computing environments depends on resolving several issues[8, 9]. One of the main challenges is resource management. It is difficult due to factors like decentralization, heterogeneity, dynamism, and variability, and it calls for effective orchestration strategies [10]. However, if the resource provisioning is not well-balanced among cloud and fog nodes, considering each request's resource demand, it changes over time [11, 12]. Effective resource management is crucial in fog computing since nodes join and exit often and dynamically [13, 14]. As such, when selecting fog nodes, resources need to be taken into account. This will ensure that only dependable nodes capable of meeting end users' QoS requirements are chosen for resource provisioning [15, 16]. Particle Swarm Optimisation (PSO) [17], Artificial Bee Colony (ABC) [18], Quantum Genetic Algorithm [19], and ABCJAYA methods were utilised in fog computing as optimization-based resource provisioning methodologies.

However, processing overhead, disregard for important attributes, and scalability are among the challenging problems in resource distribution. Thus, this study provides the optimal QoS-driven resource allocation for the fog environment. The major contributions of the research are:

- Formulated QoS- driven resource provisioning problem to minimize delay, energy and cost by considering deadlines of the user requests.
- An optimization algorithm called Enhanced Coati Optimization is proposed to address the aforementioned issue in the fog environment.
- Conducted performance analysis, the designated metrics were evaluated and the loads were varied. The outcomes show how well the suggested approach optimises and lowers cost, energy use, and delay.

Table 1. Analysis of existing methods

Reference	QoS	Optimization Technique	Advantage	Limitation
Chafi, Y., et al. [17]	Workflow time, energy	PSO	Reduces workflow time and energy consumption	Static resources are considered
Sefati et al.[18]	load balancing and QoS routing	ABC	Adjusts to changing network conditions.	WSNs focused. The applicability to fog computing scenarios has not been demonstrated.
Zanbouri, K., et al. [19]	delay, throughput	Quantum GA+ Fuzzy	High precision in task scheduling	Computational overhead, energy parameter not considered
Faizan Murtaza et al. [20]	resource utilization Latency, response time	Hybrid meta-heuristic	Latency minimization and enhancing resource utilization.	Computational complexity
K. Dubey et al.[21]	Energy, Latency and security	Heuristic-based	Reduces latency and improves energy efficiency.	Throughput and Resource utilization are not considered.

The organization of the research is as follows: Section 2 lists related works along with a problem description, and Section 3 presents a detailed proposed technique. Experimental results are detailed in Section 4, Finally, the conclusion is provided in Section 5.

2. Related Works

The existing methods, as well as their benefits and limitations, will be covered in this section. In [17], authors discussed an innovative approach based on PSO for optimizing process energy and time in diverse fog landscapes. To properly explore the solution space and adapt to fog computing resources' dynamic and unpredictable nature, their algorithm used particle collective intelligence. In their work, they compared results only with the traditional PSO for the varying tasks. In [18], authors worked to achieve a balanced load and optimize routing. They Developed the Artificial Bee Colony (ABC) and Hybrid Markov Model (MM) models. Initially, the network used load balancing amongst Cluster Heads (CHs). The Markov Model was then used to simulate the network's dynamic behavior, and the ABC approach was used to determine which cluster's best candidates for CH selection were. The Markov Model's and the ABC algorithm's computational overhead restricts their application to real-world processing. The authors of [19] suggested a fog-based multimedia transmission scheduler that uses fuzzy-based rules and quantum genetic algorithms. Fuzzy rules were based on input and output variables, offering a versatile and adaptable method for managing imprecise and uncertain data. The Quantum Genetic Algorithm was used for routing, improving population diversity, convergence speed, and accuracy. However, it was difficult to achieve the maximum PDR compared to the baseline technique. In [20], authors presented a Learning Repository Fog-Cloud (LRFC) to enhance QoS concerning processing time, response time and energy usage, an adaptable and intelligent job scheduling technique. Between IoT devices and Fog nodes, the authors have developed a smart soft layer that may be expanded to apply different kinds of learning-based policies. In [21], authors used hybrids of Cuckoo Search Optimisation (CSO) and PSO algorithms were utilized to process IoT applications and improve load balancing, latency, processing cost, and power usage. Simulations performed on a simulated dataset indicate their technique works better than contemporary baseline approaches. However, the authors only considered deadline criteria, ignoring other important factors like priority and hardware requirements. In [22], authors proposed load balancing on fog nodes.

A particle swarm optimization-dependent EDRAM was introduced to handle load balancing well; it reduces latency and task waiting times and improves user quality. In [23], the authors presented a lightweight, reliable algorithm for determining the best route and placement of services in fog topologies. There are two components to the recommended strategy. First, application-specific matrices unique to each application, such as the Analytical Hierarchy Process, are employed to select the best alternative path. Second, the optimal data paths are represented by the Pareto Frontier, which is found using PSO. This combination offers a sensible way to make judgments by simultaneously considering many objectives and application requirements. Even yet, the network lifetime becomes more challenging when energy is ignored as a routing component. Table 1 provides an analysis of existing approaches.

3. Problem Formulation

In fog environments, QoS-based routing is essential for ensuring efficient resource provisioning and meeting application-specific requirements; existing methods of QoS routing, ranging from meta-heuristic algorithms to fuzzy logic and optimization techniques, aim to enhance reliability, availability, and efficiency in fog architectures. However, these methods face challenges such as computational overhead, scalability issues, and limitations in considering critical parameters like energy consumption. To address these challenges, resource provisioning using an enhanced Coati algorithm offers a promising solution. This approach ensures efficient utilization of fog resources while meeting application-specific requirements. In the system that is being suggested, a task is defined as a user request. Let F{F_1, F_2,.., F_n} be the set of the fog nodes and T {t1,t2,..tn} be the tasks that be processed. Each task will have a deadline before which its processing must be completed. Response time (Rt) is defined as the total time spent by the task to complete its execution. In order to avoid missing the deadline, the response time of the task is considered a constraint. This is shown in the following Equation (1).

$$R_t < D_t \tag{1}$$

Initially, the algorithm analyses the incoming requests considering their deadlines, and then, to resolve the problem of suitable fog node selection, an enhanced Coati Optimization Algorithm (eCOA) is developed. The eCOA algorithm takes job deadlines into account while minimising time, energy, and cost.

3.1. System Architecture

Figure 1 presents a fog-cloud architecture that integrates both fog and cloud-based processing nodes. This architecture is structured into three distinct layers. The first layer, the Internet of Things (IoT), this layer generates user requests. Gateway is utilized between the IoT and the Fog Layer, wherein the eCOA algorithm handles the user requests. This algorithm optimizes resource allocation considering the application requirements. Fog domains and Fog Control Nodes (FCNs) make up the second layer, known as the fog layer. The FCN controls several fog nodes. The eCOA model chooses the best fog node and places the request. The third layer, the Cloud Layer, can handle and store large volumes of data, with many physical servers located within this layer.

3.2. eCOA Algorithm

The enhanced Coati Optimisation Algorithm (eCOA) and its mathematical representation are explained in detail in this section.

3.2.1. Fitness Function

The multi-objective fitness function is based on minimizing the delay, energy, and cost formulated.

Execution Delay: D is the execution delay, it is the time required to process the task and is defined by Equation (2).

$$D = \sum_{i \in T} d_{ij} \tag{2}$$

Where T is the set of tasks that forms the user request that provides the service, d_{ij} represents the processing delay of task i on fog node j and is defined by Equation (3).

$$d_{ij} = \frac{1}{P_i} * S_i \tag{3}$$

Where P_i is the processing rate (MIPS) of ith task, S_i represents the size of the data stream of ith task in Million Instructions (MI).

Energy: The fog node's energy consumption has a linear relationship with CPU utilization [24] and is represented by $E_{ij.}$ When task i is processed on node j, based on utilization of CPU(u), it is defined as in Equation (4).

$$E_{ij}(u) = c * E_{max} + (1 - c) * E_{max} * u \tag{4}$$

E_{max} is the fog node's energy consumption when operating at full capacity, c signifies the idle node's percentage of power consumption. The total energy consumption to complete a set of tasks is defined in Equation (5).

$$E = \sum_{i \in T} E_{ij} \tag{5}$$

Cost: The cost associated with each task consists of memory and processing costs, which can be defined in Equation (6).

$$C_i = \sum_{j=1}^{n} (cs_j^p * Pr_{ij} + cs_j^m * M_i) * y_{ij} \tag{6}$$

Here, y_{ij} is 0 or 1; if the fog node is available for task i, then the value is 1 otherwise 0. cs^p, cs^m are the cost of processing and memory usage on node j, respectively. M_i is the memory required for task i. The total cost for n tasks is defined as in Equation (7).

$$TC = \sum_{i=1}^{n} C_i \tag{7}$$

The fitness function for the proposed algorithm for fog node selection is implemented using the given Equation (8).

$$V = D + E + TC \tag{8}$$

Where the fitness function is denoted as V.

3.2.2. Initialization

The eCOA technique uses a population-based metaheuristic algorithm, with coatis as members of the algorithm. The decision variable values are based on each coati's location inside the search space. Coatis' position in the COA indicates a possible answer to the situation. Consequently, there are two distinct phases of updating the COA population: the exploration phase and the exploitation phase. In this, the chaotic mapping is incorporated in the COA to enhance the randomization phase, and hence, the local optimal trapping issue is solved. The optimal fog node is selected by executing the eCOA algorithm. Initializing the coati's location in the search space may be expressed mathematically as in Equation (9).

$$Y_l : y_{lm} = lbound_m + p.(ubound_m - lbound_m),\ l = 1,2, \dots N, m = 1,2,..S \tag{9}$$

Here, Y_l is the l^{th}coati's location in the search area $N$is the number of coatis,p is the probability factor between [0,1], $lbound_m, lbound_m$ is the m^{th}variable's lower and upper bound, and yl_m is the solution obtained by l^{th} coati with

$m$variables. The dimension of the solution is indicated as S. The matrix Y is utilized for the Numerical representation of coati populations based on the COA presented in Equation (10).

$$Y = \begin{bmatrix} Y_1 \\ \vdots \\ Y_l \\ \vdots \\ Y_N \end{bmatrix}_{N\times S} = \begin{bmatrix} y_{1,1} \cdots y_{1,m} \cdots y_{1,S} \\ \vdots \ddots \vdots \\ y_{l,1} \cdots y_{l,m} \cdots y_{l,S} \\ \vdots \ddots \vdots \\ y_{N,1} \cdots y_{N,m} \cdots y_{N,S} \end{bmatrix}_{N\times S} \tag{10}$$

Equation (11) is employed to evaluate a broad spectrum of values for the problem's objective function, influenced by the arrangement of potential solutions within the choice variables.

$$V = \begin{bmatrix} V_1 \\ : \\ V_l \\ : \\ V_N \end{bmatrix}_{N\times 1} = \begin{bmatrix} V(Y_1) \\ : \\ V(Y_l) \\ : \\ V(Y_N) \end{bmatrix}_{N\times 1} \tag{11}$$

Here, the target solution and the evaluated solution are indicated as V and V_l respectively. The optimal member position and candidate solution may vary as the algorithm iterates.

3.2.3. Exploration Phase

The exploration phase in the Iguana COATI algorithm is characterized by a hunting and attacking strategy. The location of the coatis' when they emerge from the branch can be represented as in Equation (12).

$$Y_l^{e1}: y_{lm}^{e1} = y_{lm} + p.(Ig_m - W.Y_l) for l = 1,2\ldots\ldots \left\lfloor \frac{N}{2} \right\rfloor ,\text{m}=1, 2\ldots\text{S} \tag{12}$$

Where el specifies the exploration phase of the algorithm and $W$is the integer chosen within the set {1,2}. The prey, i.e. iguana identified by the coati, is denoted as $I_{gm.}$When the prey drops to the land, it is randomly placed within the search area. The ground-based coatis moves within the search area using Equations (13) and (14) according to this random position.

$$Ig^F: Ig_m^F = lbound_m + p.(ubound_m - lbound_m, m = 1,2,\ldots., S) \tag{13}$$

$$Y_l^{e1}: y_{l,m}^{e1} = \begin{cases} Y_{l,m} + p.(Ig_m^F - W \cdot y_{l.m}), V_{Ig^F} < V_l, \\ Y_{l,m} + p.(y_{l.m} - Ig_m^F), else, \end{cases}$$

For

$$l = \left\lfloor \frac{N}{2} \right\rfloor + 1, \left\lfloor \frac{N}{2} \right\rfloor + 2, \ldots . N\text{m} = 1, 2\ldots \text{S} \tag{14}$$

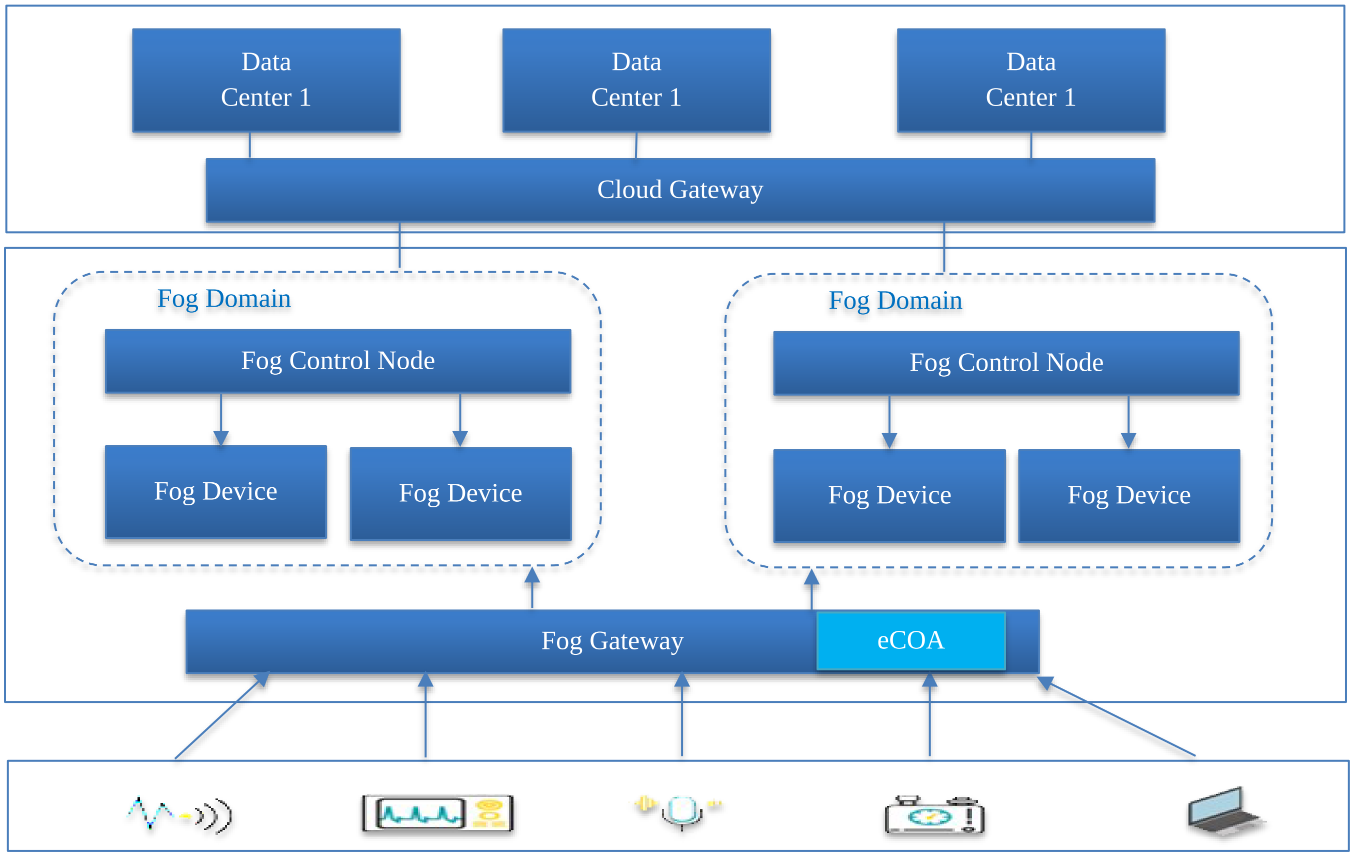

Fig. 1 System architecture

The new location for each coati is considered acceptable only when it enhances the value of the objective function. This is represented in the Equation (15).

$$Y_l = \begin{cases} Y_l^{e1}\ , V_l^{e1} \ < V_l, \\ Y_l, else. \end{cases} \tag{15}$$

Here, y_l^{e1} is the solution identified by l^{th}coati, $y_{l,m}^{e1}$ is them^{th}dimension value, V_l^{e1} refers to the objective function value and $p$is the real number. Also, Ig_m^F signifies the randomly selected location of prey on the ground in its mth dimension. Wact as an integer.

3.2.4. Exploitation Phase

This step is designed to enhance the coatis' positions within the search area while avoiding predators. Using Equations (16) and (17) a random position is generated near each coati's location.

$$lbound_m^{local} = \frac{lbound_m}{T}, ubound_m^{local} = \frac{ubound_m}{T} \tag{16}$$

Where, $T = 1,2, t$

$$Y_l^{e2}: y_{l,m}^{e2} = y_{l,m} + (1-2p).(lbound_m^{local}$$

$$+p.(ubound_m^{local} - lbound_m^{local})) \tag{17}$$

Where, l=1, 2… N, m =1, 2… S

If the objective function value is enhanced, the newly computed position is expressed in Equation (18).

$$Y_l = \begin{cases} Y_l^{e2}, V_l^{e2} < V_l; \\ Y_l, else, \end{cases} \tag{18}$$

Here, $Y_l^{e2}$is the new position of l^{th}coati. According to the COA's second phase, itsm^{th} dimension is denoted as $Y_{l,m}^{e2}$, the objective function value is denoted as V_l^{e2}, where $p$is a random value between 0 and 1, t is an iteration counter, $lbound_m^{local}$ and $ubound_m^{local}$ denotes the m^{th} decision variable's lower and upper limits and thez^{th} variable has lower and upper bounds which is denoted as $lbound_m$and $ubound_m$respectively.

3.2.5. Enhanced Coati Optimization Algorithm

We introduced Levy Flight (LF) to achieve the balance between exploration and exploitation. This is achieved with the levy flight steps. These steps are based on Levy distribution.

$$Y_l^{LF} = Y_l + p.LF(\beta) \tag{19}$$

In the Equation (19), Y_l^{LF} is the lth coati's latest location, dot(.) denotes dot product, p is either 0 or 1, LF(β) is defined as in the following Equation (20).

$$LF(\beta) = \ 0.01 * \frac{u}{|v|^{1/\beta}} \tag{20}$$

Here, u and v represents numbers drawn from the normal distribution and 1< β<2. After the location of lth coati Y_l is modified using Equation (19), let Y_l^*, the following LF incorporated equation can be used to obtain the novel candidate solution as in the following Equation (21).

$$Y_l^{LF} = Y_l^* + p.LF(\beta) \tag{21}$$

$$Y_l^{i+1} = \begin{cases} Y_l^F,\ fitness(Y_l^{LF}) > fintness(Y_l^*) \\ Y_l^*,\ \ otherwise \end{cases} \tag{22}$$

Thus, using the solution acquired by the ICOA, the tasks are routed in the fog layer for processing the request through the Edge devices.

Pseudo-code of proposed eCOA algorithm

Algorithm: Pseudo-code of eCOA algorithm

```
Start
1. Set the maximal iterations (T)and the number of coatis.(N)
2. Evaluate the objective function of the initial population. Initialize all the positions of the coatis.
3. For
   The Prey's position is updated according to the greatest population density.
**Phase 1:** Exploration stage:
4. For
5. Compute and update the new position of  coati using Equations (12) and (15).
6. End for
7. For
8. Compute random location and find the optimal location for xth coati using Equations (13) and (14)
9. Update the location of xth coati with the Equation (15)
10. End for
**Phase 2 :** Exploitation stage
11. Local bounds are calculated using Equation(16).
12. For
13. Compute and update novel location of xth coati using Equations (19) and (20).
14. End for
15. For x<T
16. Execute levy method using Equations (21) and (22)
17. Return Yl if the limit is reached maximum.
18. End For
19. Compute the fitness
20. Update Y with the optimum value
21. r=r+1
22. Return optimum solution
23. End for
   Output for the eCOA algorithm
24. End
```

4. Result and Discussion

The proposed eCOA is implemented in iFogSim and is analyzed based on various measures. The topology of the fog-cloud architecture presented in Section 3.1 is taken into consideration by extending iFogSim. Application deadlines and deployment times are considered by modifying the Application and AppModule classes. FogDevice class is extended to represent EdgedeviceCOA to act as a fog node. To evaluate the efficiency of eCOA, 1000 tasks to be processed are generated with varying storage capacity and deadlines. eCOA chooses the optimal fog node to place these tasks. The proposed model is compared with existing methods like ABC, GA, PSO, Coati Optimization algorithm (COA) [25], and ABCJAYA [26] to indicate the proposed method's superiority.

4.1. Delay

The delay of the resource provisioning in fog environment methods refers to the total time taken from submitting a workload request to its completion and delivery to the intended destination. Figure 2 illustrates the delay analysis. The proposed eCOA algorithm checks the deadlines of the applications and places the tasks accordingly in the fog environment, potentially impacting delay. The end-to-end delay by the eCOA, considering 200 tasks, is 53.68ms, which is 8.85%, 15.98%,17.44%, 24.62% and 42.16% better than ABCJAYA, COA, ABC, GA, and PSO methods.

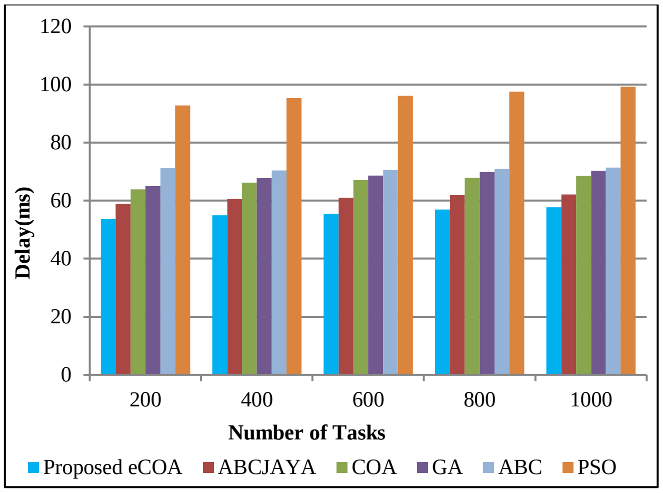

Fig. 2 Analysis based on delay

4.2. Energy Consumption

Figure 3 illustrates the energy usage analysis. As the task increases, the energy consumption increases. However, the proposed method accomplished a superior outcome for all the cases.

The energy consumption by the eCOA considering 200 tasks is 26.12Kwh, which is 7.83%, 16.31%,25.58%, 36.29% and 43.3% better than ABCJAYA, COA, ABC, GA, and PSO methods.

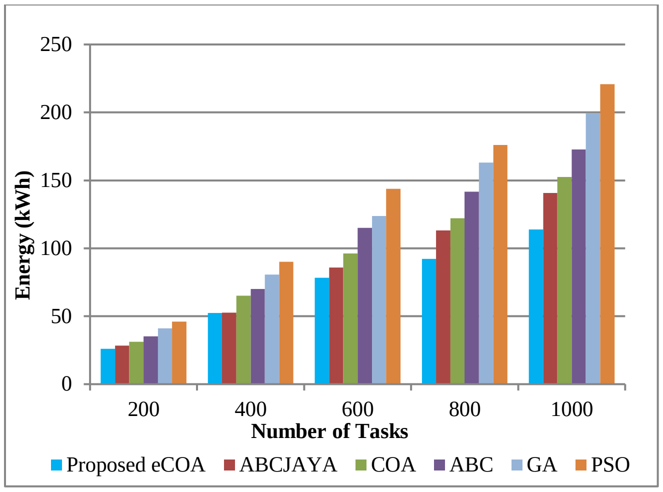

Fig. 3 Analysis based on energy consumption

4.3. Cost Analysis

Figure 4 shows the cost analysis. In the proposed model, consideration of cost as the fitness factor of the optimization algorithm assists the algorithm in choosing the optimal best route with minimal cost. The cost by the eCOA, considering 200 tasks, is 0.52$, which is 7.14%, 14.75%,24.64%, 34.18% and 39.53% better than ABCJAYA, COA, ABC, GA, and PSO methods.

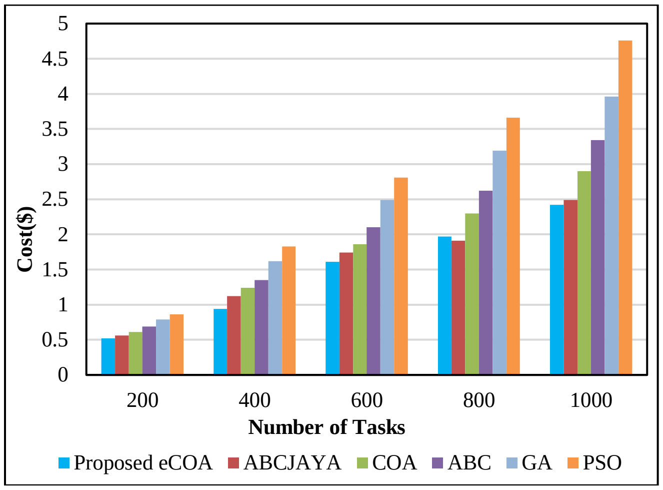

Fig. 4 Analysis based on cost

4.4. Processing Time

The Processing time is the amount of time a task takes to complete its execution. It includes the time it enters the system until it is completed. The processing time-based analysis is presented in Figure 5. The proposed method gives priority to the tasks based on the deadlines. The eCOA is utilized to select suitable fog nodes for workload execution. The algorithm aims to minimize request execution delay by selecting fog nodes that are optimally located and have sufficient resources available. Thus, the proposed method acquired minimal processing time compared to the existing methods.

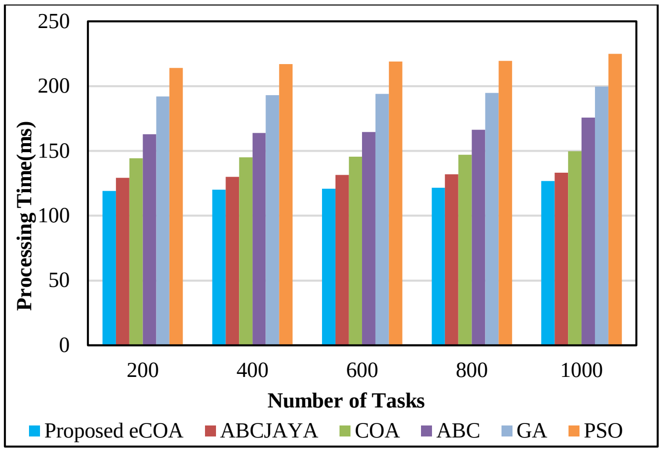

Fig. 5 Analysis based on processing time

The execution time by the eCOA, considering 200 tasks, is 119.06ms, which is 7.8%, 17.45%,26.87%, 38.01% and 44.4% better than ABCJAYA, COA, ABC, GA, and PSO methods.

4.5. Comparative Analysis

The comparison of the proposed method with the existing methods based on the best case is shown in Table 2. The proposed method takes into account application QoS requirements during task processing, enabling it to effectively minimize delay and cost.Selecting fog nodes with adequate resources further reduces processing time and energy consumption. As a result, the proposed eCOA algorithm demonstrates superior performance compared to existing methods.

Table 2. Comparison based on best-case

Methods	Delay (ms)	Energy (kWh)	Cost ($)	Processing Time(ms)
Proposed eCOA	53.68	26.12	0.52	119.06
ABCJAYA	58.89	28.34	0.56	129.14
COA	63.89	31.21	0.61	144.22
ABC	65.02	35.1	0.69	162.8
GA	71.21	41	0.79	192.061
PSO	92.81	46.07	0.86	214.122

5. Conclusion

The proposed research introduced a novel resource provisioning algorithm for enhancing the QoS of the model. This work considers the deadlines of the tasks and applies the eCOA algorithm for efficient resource provisioning. The eCOA has improved overall system performance, enabling fog environments to adapt more effectively to changing workload demands. The proposed eCOA is evaluated in terms of delay, energy consumption, execution time, and cost, and it acquired the value of 53.68ms, 26.12Kwh,119.06ms, and 0.52$, respectively. Compared to current techniques, the proposed algorithm reduces delay, cost, and energy, making it suitable for real-time Internet of Things applications.

In the future, prioritization of the tasks based on their requirements can be done for efficient resource provisioning. Also, the integration of machine learning techniques will be designed to enhance the adaptability and intelligence of fog computing environments.

References

[1] Samodha Pallewatta, Vassilis Kostakos, and Rajkumar Buyya, "QoS-Aware Placement of Microservices-Based IoT Applications in Fog Computing Environments," *Future Generation Computer Systems*, vol. 131, pp. 121-136, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[2] Chang Liu et al., "Solving the Multi-Objective Problem of IoT Service Placement in Fog Computing Using Cuckoo Search Algorithm," *Neural Processing Letters*, vol. 54, pp. 1823-1854, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[3] Mahboubeh Salimian, Mostafa Ghobaei-Arani, and Ali Shahidinejad, "An Evolutionary Multi-Objective Optimization Technique to Deploy the IoT Services in Fog-Enabled Networks: An Autonomous Approach," *Applied Artificial Intelligence: An International Journal*, vol. 36, no. 1, pp. 1-34, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[4] Olena Skarlat et al., "Optimized IoT Service Placement in the Fog," *Service Oriented Computing and Applications*, vol. 11, pp. 427-443, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[5] Kirandeep Kaur, Arjan Singh, and Anju Sharma, "A Systematic Review on Resource Provisioning in Fog Computing," *Transactions on Emerging Telecommunications Technologies*, vol. 34, no. 4, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[6] Zhijun Zhang, Hui Sun, and Hajar Abutuqayqah, "An Efficient and Autonomous Scheme for Solving IoT Service Placement Problem Using the Improved Archimedes Optimization Algorithm," *Journal of King Saud University - Computer and Information Sciences*, vol. 35, no. 3, pp. 157-175, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[7] Suresh Kumar Srichandan et al., "A Secure and Distributed Placement for Quality of Service-Aware IoT Requests in Fog-Cloud of Things: A Novel Joint Algorithmic Approach," *IEEE Access*, vol. 12, pp. 56730-56748, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[8] Shuaibing Lu et al., "QoS-aware Online Service Provisioning and Updating in Cost-Efficient Multi-Tenant Mobile Edge Computing," *IEEE Transactions on Services Computing*, vol. 17, no. 1, pp. 113-126, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[9] Abdelhamied A. Ateya et al., "Edge Computing Platform with Efficient Migration Scheme for 5G/6G Networks," *Computer Systems Science and Engineering*, vol. 45, no. 2, pp. 1775-1787, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[10] Defu Zhao, Qunying Zou, and Milad Boshkani Zadeh, "A QoS-Aware IoT Service Placement Mechanism in Fog Computing Based on Open-Source Development Model," *Journal of Grid Computing*, vol. 20, no. 12, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[11] Soroush Hashemifar, and Amir Rajabzadeh, "Optimal Service Provisioning in IoT Fog-Based Environment for QoS-Aware Delay-Sensitive Application," *Computers and Electrical Engineering*, vol. 111, no. B, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[12] Mirsaeid Hosseini Shirvani, and Yaser Ramzanpoor, "Multi-Objective QoS-Aware Optimization for Deployment of IoT Applications on Cloud and Fog Computing Infrastructure," *Neural Computing and Applications*, vol. 35, pp. 19581-19626, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[13] Amir Hossein Mokabberi, Aliakbar Iranmehr, and Mehdi Golsorkhtabaramiri, "A Review of Energy-efficient QoS-aware Composition in the Internet of Things," *8th International Conference on Technology and Energy Management (ICTEM)*, Iran, pp. 1-6, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[14] Meenu Vijarania et al., "Energy Efficient Load-Balancing Mechanism in Integrated IoT-Fog-Cloud Environment," *Electronics*, vol. 12, no. 11, pp. 1-13, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[15] Resul Das, and Muhammad Muhammad Inuwa, "A Review on Fog Computing: Issues, Characteristics, Challenges, and Potential Applications," *Telematics and Informatics Reports*, vol. 10, pp. 1-20, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[16] K. Bhargavi, B. Sathish Babu, and Sajjan G. Shiva, "Type-2-Soft-Set Based Uncertainty Aware Task Offloading Framework for Fog Computing Using Apprenticeship Learning," *Cybernetics and Information Technologies*, vol. 23, pp. 38-58, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[17] Saad-Eddine Chafi et al., "Novel PSO-Based Algorithm for Workflow Time and Energy Optimization in a Heterogeneous Fog Computing Environment," *IEEE Access*, vol. 12, pp. 41517-41530, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[18] Seyed Salar Sefati, Mehrdad Abdi, and Ali Ghaffari, "QoS-Based Routing Protocol and Load Balancing in Wireless Sensor Networks Using the Markov Model and the Artificial Bee Colony Algorithm," *Peer-to-Peer Networking and Applications*, vol. 16, no. 3, pp. 1499-1512, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[19] Kouros Zanbouri et al., "A New Fog-Based Transmission Scheduler on the Internet of Multimedia Things Using a Fuzzy-Based Quantum Genetic Algorithm," *IEEE MultiMedia*, vol. 30, no. 3, pp. 74-86, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[20] Faizan Murtaza et al., "QoS-Aware Service Provisioning in Fog Computing," *Journal of Network and Computer Applications*, vol. 165, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[21] Kalka Dubey, S.C. Sharma, and Mohit Kumar, "A Secure IoT Applications Allocation Framework for Integrated Fog-Cloud Environment," *Journal of Grid Computing*, vol. 20, no. 1, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[22] D. Baburao, T. Pavankumar, and C.S.R. Prabhu, "Load Balancing in the Fog Nodes Using Particle Swarm Optimization-Based Enhanced Dynamic Resource Allocation Method," *Applied Nanoscience*, vol. 13, pp. 1045-1054, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[23] Nerijus Morkevicius, Agnius Liutkevicius, and Algimantas Venckauskas, "Multi-Objective Path Optimization in Fog Architectures Using the Particle Swarm Optimization Approach," *Sensors*, vol. 23, no. 6, pp. 1-20, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[24] Anton Beloglazov, and Rajkumar Buyya, "Optimal Online Deterministic Algorithms and Adaptive Heuristics for Energy and Performance Efficient Dynamic Consolidation of Virtual Machines in Cloud Data Centers," *Concurrency and Computation: Practice and Experience*, vol. 24, no. 13, pp. 1397-1420, 2012. [CrossRef] [Google Scholar] [Publisher Link]

[25] Mohammad Dehghani et al., "Coati Optimization Algorithm: A New Bio-Inspired Metaheuristic Algorithm for Solving Optimization Problems," *Knowledge-Based Systems*, vol. 259, pp. 1-43, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[26] Usha Vadde, and Vijaya Sri Kompalli, "Energy Efficient Service Placement in Fog Computing," *PeerJ Computer Science*, vol. 8, pp. 1-16, 2022. [CrossRef] [Google Scholar] [Publisher Link]