International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7260
***Article Received**: 30 March 2023 **Revised**: 28 April 2023 **Accepted**: 14 June 2023*

Dynamic Offloading Technique for Latency-Sensitive Internet of Things Applications using Fog Computing

Priya Thomas[1], Deepa V Jose[2]
[1]Research Scholar, Dept of Computer Science
[2]Associate Professor, Dept of Computer Science
CHRIST (Deemed to be University)
Bangalore, India
[1]priya.thomas@res.christuniversity.in
[2]deepa.v.jose@christuniversity.in

Abstract—Internet of Things (IoT) has evolved as a novel paradigm that provides com-putation power to different entities connected to it. IoT offers services to multiple sectors such as home automation, industrial automation, traffic management, healthcare sector, agriculture industry etc. IoT generally relies on cloud data centers for extended analytics, processing and storage support. The cloud offers highly scalable and robust platform for IoT applications. But latency sensitive IoT applications suffer delay issues as the cloud lies in remote location. Edge/fog computing was introduced to overcome the issues faced by delay-sensitive IoT applications. These platforms lie close to the IoT network, reducing the delay and response time. The fog nodes are usually distributed in nature. The data has to be properly offloaded to available fog nodes using efficient strategies to gain benefit from the integration. Differ-ent offloading schemes are available in the literature to overcome this prob-lem This paper proposes a novel offloading approach by combining two effi-cient metaheuristic algorithms, Honey Badger Algorithm (HBA) and Fla-mingo Search Algorithm (FSA) termed as HB-FS algorithm. The HB-FS is executed in an iterative manner optimizing the objective function in each it-eration. The performance evaluation of the proposed approach is done with different existing metaheuristic algorithms and the evaluations show that the proposed work outperforms the existing algorithms in terms of latency, response time and execution time. The methodology also offers better degree of imbalance with proper load balancing under different conditions.

Keywords- Task Offloading, Internet of Things, Latency sensitive IoT, Fog Computing, Quality of Service, Cloud Computing.

I. INTRODUCTION

Internet of things (IoT) has emerged as a promising technology within a short span of time. The IoT has made life smarter and more comfortable. It has drastically transformed the way different applications operate. The old-fashioned cities and homes have been made smart with an improved Quality of Service (QoS) due to the integration of IoT. Due to millions of connected applications, IoT generates huge amounts of data every second. IoT nodes are energy-constrained and it generally depends on cloud platforms to provide support for the storage and analysis of generated data. Cloud data centers offer robust and reliable services to connected applications based on pay per use policy. Moreover, the cloud also helps to achieve the QoS and quality of experience (QoE) requirements of the applications [1]. Even though lots of applications benefit from this extension, cloud being at a remote location suffers from few drawbacks. The problem of single point of failure and higher response time due to congestion in the path has to be addressed to make the integration more acceptable. Fog computing was designed as an extension to cloud computing where processing happens near the edge of the connected network reducing the response time and latency-related issues. The fog is distributed in nature solving the problem of single point of failure. The terminology fog computing lies synonymous with the terms: edge computing, mist computing, micro cloudlets etc.

Fog computing was designed to support real-time applications with better reliability and improved security. Fog also supports better load balancing and faster responses as it lies in the edge of the network. Fog improves the scalability and data confidentiality providing organizations a better control over their resources lying within the premises of the organization network [2]. Therefore, delay-sensitive ap-plications can rely on fog computing for task offloading with minimum drainage of battery life. This has framed the IoT-Fog-Cloud three-tier architecture where IoT devices which generates input data forms the lower most layer passes to intermediate fog layer which supports distributed processing of the received data. The upper most layer is the cloud layer which supports extended storage, processing and analytics. This three-tier architecture has gained wide popularity, especially for time-sensitive IoT applications. But the fog layer also suffers from few drawbacks which needs to be addressed to make the integration more powerful. The fog being

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7260
***Article Received**: 30 March 2023 **Revised**: 28 April 2023 **Accepted**: 14 June 2023*

distributed in nature requires strong algorithms to offload the incoming data to the available fog nodes. The storage and processing capacity of individual fog nodes also has to be considered while designing the proper framework. The task deadline, the avail-ability of fog nodes, the conditions of overload and underload, etc. need to be ad-dressed while integrating fog into the IoT application. Since the distribution of tasks to fog nodes, which is also termed offloading is an active area of research, this work focuses on designing an effective mechanism for offloading the tasks to the fog lay-er with the support of efficient meta-heuristic algorithms.

There are several techniques discussed in the literature to address the offloading issues. The survey on these techniques revealed few research gaps in this area. The common research gaps are as follows: Most of the offloading techniques focus on improving one or two parameters only. But this will not significantly improve the overall QoS. More parameters have to be considered to improve the overall performance of the IoT applications. Another research gap identified is the lack of novelty in chosen techniques for offloading. Most of the works repeatedly use the same algorithms with minor modifications. Novel algorithms need to be explored and experimented to reduce the complexity. One approach to solve the offloading problem is to consider it as an optimization problem. Meta-heuristic algorithms are wide class of algorithms that provide feasible solutions to many optimization problems. These algorithms also need to be experimented more. By considering the identified re-search gap, this work focuses on designing an efficient meta-heuristic-based task offloading framework that tries to minimizes the latency, response time, communication cost and execution time with better load balancing.

The proposed offloading model is designed using a combination of HBA [3] and FSA [4] algorithms (HB-FS algorithm). The two algorithms were evaluated and features were analyzed and conclusions were derived in our previous work. The HBA is an algorithm which guarantees better convergence speed, output stability and can be used for solving optimization problems with complex search-space. But HBA also sometimes suffers from the problem of getting trapped in local optima resulting in earlier convergence [5]. The FSA algorithm is proved to have strong exploration capabilities without getting trapped in local optima. This feature is utilized in the proposed model to make the solution more accurate. The FSA and HBA algorithms are modified and combined to suit the requirements for our offloading problem. The FSA iterates on the output of digging phase of honey badger algorithm and generates the final optimized output. The numerical analysis of the proposed model shows better response time, reduced latency, execution time and better degree of imbalance compared to algorithms such as Particle Swarm Optimization, Grey Wolf Optimization, Slap Swarm optimization, Genetic Algorithm, Modified Ant Colony Optimization Algorithm etc.

The major contributions of the proposed model are as follows:

- The task offloading problem in the IoT- Fog environment is formulated as a multi-objective optimization problem.
- The identified problem is NP-hard and to effectively solve the problem, a combined approach, HB-FS is proposed which is a combination of two major optimization algorithms: flamingo search algorithm (FSA) and honey badger algorithm (HBA).
- The solution for proposed problem works by optimizing different parameters such as latency, response time, communication cost, resource utilization, and execution time.
- Extensive performance evaluation of the proposed approach is done under different scenarios to prove the efficiency of proposed model over the compared approaches.

The remainder of the paper is organized as follows: section 2 surveys the existing techniques in literature, section 3 elaborates on the proposed methodology, section 4 discusses the results of performance evaluation and section 5 concludes the paper.

II. LITERATURE REVIEW

The offloading problem in IoT-Fog-Cloud integration is addressed using different schemes in the literature. This section discusses few recent advancements in the field. A computational offloading scheme based on Q learning in a Fog-Cloud environment is introduced by S. Aljanabi et al. The paper models the offloading problem as a Markov model and finds the solution to the model using the reinforcement learning approach. Numerical results show that the work provides better balancing of workload with reduced delay [2]. The issues such as communication overhead and decision-making time need to be addressed to make the integration more powerful.

The offloading problem in edge-cloud environment is addressed using a hierarchical game model by Mingyue Yu et al. The paper discusses a cost-effective solution for the offloading problem with comparatively better load balancing [6]. Reducing the task delay is required to improve the performance of the application further. The service requests from the IoT devices are modelled as an optimization problem in [7]. The problem being NP-hard uses Genetic algorithm to solve it. The performance evaluations show that the work offers lower latency compared to existing techniques.

Baek, J et al. use reinforcement learning-based approach to minimize the task processing time and overall overloading probability. The mechanism helps incoming tasks to be directed to the appropriate fog node which guarantees the QoS

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7260
***Article Received**: 30 March 2023 **Revised**: 28 April 2023 **Accepted**: 14 June 2023*

requirements. The results show that the paper achieves better convergence speed and complexity [8]. A neuro-fuzzy model based offloading approach is modelled in [9] where offloading is made more secure with machine learning approach. The offloading problem is solved using Particle Swarm Optimization technique. The numerical results shows that the approach improves security.

Genetic algorithm based offloading scheme is proposed by M.Abbasi et al. The paper focuses on reducing the energy consumption and delay between the fog nodes and the cloud. The multi-objective optimization model uses NSGA II for solving the offloading problem [10]. M.Bala et al. proposes proximity algorithm for placing the data to nearest available fog nodes and Cluster algorithm [11] to reduce the band-width consumption in the network. The experimental evaluations shows that the work offers lesser latency but needs to work on more parameters for optimization.

Resource allocation problem in edge-cloud model is formulated as a stochastic game model in [12]. The formulated problem is solved using NSGA II algorithm. The results show that the proposed model is energy efficient and cost effective. Healthcare sector is one of the major applications of IoT. A secure offloading scheme for healthcare sector is proposed by V.Meena et al. The paper uses algorithms such as optimal service offloader and trust assessment [13] to identify the appropriate fog nodes and to offload data in a secure manner.

The power consumption and delay issues in the fog cloud integration is addressed by M. Keshavarznejad et al. The paper formulates a multi objective optimization problem for task offloading and uses two meta-heuristic algorithms such as NSGA II and Bees algorithm [14] to solve the problem. The numerical analysis show that the model offers lesser energy consumption and delay. Smart Ant Colony based optimization technique is proposed in [15] to solve the computational offloading problem. The paper formulates the offloading problem as a multi-objective optimization problem and uses meta-heuristic approach to solve it. The algorithm offers reduced complexity with better load balancing.

QoS aware task offloading scheme is proposed by P. Kauret et. al. The paper uses Smart Flower Optimization Algorithm to improve the QoS requirements of the fog nodes. The parameters such as task deadline and budget constraints are taken into consideration for optimization [16]. The performance evaluations shows that the proposed approach works well with small sized workload.

An extended multi-layer reinforcement learning based task offloading framework is proposed in [17]. The framework considers tradeoff between energy consumption and execution time to generate a feasible strategy which minimizes both. The numerical evaluations shows that the proposed approach offers better performance under complex and uncertain situations.

Yakubu I.Z. et al uses m modified Harris-Hawks Optimization technique to address the task offloading problem in fog layer. The paper follows a priority-based task scheduling with the aim to minimize the execution time, latency, cost etc. and to enhance the resource usage [18]. The experimental evaluations shows that the pro-posed method effectively balances the load over the compared approaches and im-proves the performance of the system.

The survey on recent works in computational offloading algorithms shows that most of the approaches focus on one or two parameters only. More parameters need to be optimized for improving the QoS requirements. The survey of existing approaches also show that novel optimization algorithms are least explored with very few exceptions.

III. PROPOSED MODEL FOR TASK OFFLOADING

The proposed model is built on the IoT-Fog-Cloud three tier architecture. The three-tier architecture has IoT devices at lower most layer and distributed fog nodes in the intermediate layer. The intermediate layer processes the incoming IoT requests. The cloud layer is used for permanent storage. The three-tier model is depicted in Fig 1. below.

The IoT devices are capable of generating different types of data which are considered as requests to the fog layer. The data generated from IoT devices in the lower most layer is assumed to be time constrained. The data will be forwarded to the intermediate fog layer through wireless communication medium. Each fog node will have a maximum capacity. It can accept the requests till the maximum limit is reached. The overload and underload fog nodes will be continuously monitored, and load balancing will be done based on the chosen algorithms. The problem formulation considers the main objectives that as network latency, energy consumption, resource utilization, total load and execution time.

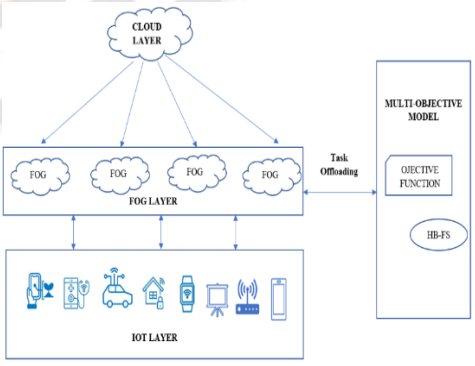

Fig. 1. Architecture of proposed IoT-Fog-Cloud model for optimal task offloading

The workflow of the proposed model is as follows: Initially, the offloading problem is formulated as a multi-objective

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7260
***Article Received**: 30 March 2023 **Revised**: 28 April 2023 **Accepted**: 14 June 2023*

optimization problem. To solve the problem, an effective strategy is proposed with an aim to optimize different parameters such as network latency, response time, communication cost, resource utilization and execution time.

As the identified problem is NP-hard, a combined approach using two recent meta-heuristic algorithms, the Honey Badger Algorithm (HBA) [3] and the Flamingo Search Algorithm (FSA) [4] is proposed. The HB-FS algorithm works by optimizing the solution in every iteration generating the final optimized output. Finally, the experimental evaluations are done under different scenarios to prove the efficiency of the proposed approach.

A. Proposed System model

The system model used for task offloading in IoT-Fog-Cloud architecture is explained in this section. The proposed system is basically designed to reduce the delay and latency issues in IoT applications that are latency sensitive and to improve the overall performance of IoT-Fog-Cloud architecture.

Formulation of computational model

The IoT-Fog-Cloud computational model uses different terminologies and variables for formulation of the model as given below:

- S_{ij}: Sensor nodes used in lower layer where $j=1, 2\ldots. n.$
- fg_{ij}: Fog nodes in the middle layer where $j=1,2\ldots n.$
- λ_{ij}: rate of data generated by the sensor nodes
- L_{jk}: Network latency between S_{ij} and fg_{ij}
- L_{jc}: Network latency between Fog layer and cloud layer
- $FgCCost_{jk}$: Communication Cost among S_{ij} and associated fog node fg_{ij}
- Ds_j: Size of data
- B_l: Bandwidth of the local network
- $CloudCCost_{jk}$: Communication cost among fg_{ij} and *Cloud*
- L_{jc}: Network latency among cloud and fg_{ij}
- B_c: Bandwidth of the cloud
- $CCost_{jk}$: Total communication cost
- μ_{jk}: Service rate
- ST_{jk}: Task offloading service time
- U_{ij}: Resource utilization
- R_{jk}: Response time

The IoT-Fog-Cloud model has 'n' number of IoT sensors S_{i1}, S_{i2}, …. S_{im}, where each sensor node, S_{ij} will be generating the data at rate λ_{ij}. The data generated by IoT layer moves to the fog layer which has distributed set of fog nodes fg_{i1}, fg_{i2} …, fg_{in}. Different connected sensors S_{ij} will generate the data based on Poisson distribution with the data arrival rate λ_{ij}. The generated data will be time constrained and will be processed at the fog nodes and will be forwarded to the remote cloud for extended storage processing and analysis. The total communication cost can be calculated as the sum of communication cost among the S_{ij} and the associated fog node fg_{ij} and communication cost among fog node fg_{ij} and the Cloud as represented by (1).

$$CCost_{jk} = CloudCCost_{jk} + FogCCost_{jk} \quad (1)$$

where, $FgCCost_{jk}$ and $CloudCCost_{jk}$ can be computed using the equations (2) and (3). $FgCCost_{jk}$ is computed as the sum of L_{jk} (network latency) among S_{ij} and fg_{ij} and transfer time of data among S_{ij} and fg_{ij}

$$FgCCost_{jk} = L_{jk} + \frac{Ds_j}{B_l} \quad (2)$$

The cloud cost is the sum of cloud latency L_{jc} and transfer time of data among cloud and fg_{ij}.

$$CloudCCost_{jk} = L_{jc} + \frac{Ds_j}{B_c} \quad (3)$$

where B_c is the bandwidth of cloud.

The service time of task offloading of S_{ij} to fg_{ij} is computed based on the *M/M/1* queuing model as given in (4).

$$ST_{jk} = \frac{1}{\mu_{jk} - \lambda_{jk}} \quad (4)$$

where, μ_{jk} is the service rate. The total latency will be computed as a sum of communication latency and computation latency which is a function of time. The computation latency is also influenced by the underlying offloading algorithm policies such as decision-making time, time taken to balance the tasks among available fog nodes and number of tasks being offloaded to each fog nodes.

The resource utilization (U_{ij}) value of fog nodes is computed as summation of resource utilization of different offloaded tasks by connected sensors as given in (5).

$$U_{ij} = \sum_c \sum_{k \epsilon c} y_{ij} \times \frac{\lambda_{ij}}{\mu_{ij}} \quad (5)$$

Here, $yij=1$, when S_{ij} tasks are offloaded to fg_{ij}.

The total response time is computed as:

$$R_{jk} = CCost_{jk} + ST_{jc} \quad (6)$$

The load of connected nodes can be evaluated using (7) and (8) as given below:

$$R_k = \sum_k R_{kc} / (1 - \mu_{jc}) \quad (7)$$

$$Load_k = 1 - \frac{(R_k - R_{average})}{\sum_k R_k} \quad (8)$$

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7260
***Article Received**: 30 March 2023 **Revised**: 28 April 2023 **Accepted**: 14 June 2023*

B. Proposed task offloading algorithm

The task offloading problem is designed to effectively distribute the incoming IoT tasks to the available fog nodes. The fog nodes chosen by the tasks should be capable of meeting the QoS requirements. The complexity of this problem will exponentially increase with the increase in number of sensors and fog nodes. So, to solve this problem we need optimization algorithms which are efficient as well as less complex. Meta-heuristic algorithms are wide class of algorithms designed to solve complex optimization problems. According to No-Free-Lunch theorem, none of the optimization algorithms are capable of solving all the problems. When two algorithms are combined, we can obtain the benefits from both optimization algorithms. This work proposes a hybridized approach by combining the essential features of HBA and FSA algorithms. The output obtained from HB-FS algorithm proves to be better solution with reduced latency, average response time, execution time and standard deviation.

Proposed HB-FS Algorithm

The proposed HB-FS algorithm for task offloading.

1. Set parameters like C_j, β, M, number of nodes, max, flamingos and t_m.
2. Iterate digging phase of HBA.
3. Initialize the Flamingo population using the sample generated in step 2.
4. Initialize foraging and migrating characteristics of flamingos.
5. Perform position update using foraging and migration phase.
6. Evaluate fitness and update the best position in each iteration.
7. Repeat the steps 4-7 until desired solution is obtained.
8. End.

Honey Badger Algorithm (HBA) is a recent optimization algorithm designed to model the predatory behavior of Honey Badger, an intelligent mammal found in areas like India's subcontinent, Africa, and southwest Asia. The animal is well known for its predatory behavior and fear-less nature. The HBA algorithm is designed in two phases: digging phase and honey phase. The honey badger identifies prey using the smell intensity of prey and dig holes to catch the prey. During honey phase, the honey badger takes the help of honey bird to identify honey. The digging phase is utilized in the proposed model.

The HBA algorithm is initialized using (9)

$$y_j = LL_j + rand_1 + (UL_j - LL_j) \quad (9)$$

where $rand_1$ is a random number ranging between 0 and 1. The j^{th} position of honey badger is represented as y_j, and the lower and upper bounds of the entire search space is given as LL_j and UL_j.

The smell intensity of the prey I_n is updated using (10) where, $rand_2$ is a random number between 0 and 1.

$$I_n = rand_2 \times \frac{R}{4\pi D_j^2} \quad (10)$$

where, $$R = (y_j - y_{j+1})^2 \quad (11)$$

$$D_j = y_{prey} - y_j \quad (12)$$

Here, the focusing power is represented as R, distance between prey and j^{th} HB is given as D and the global best position of prey is given as y_{prey}.

Density factor (α) updations: This factor is responsible for controlling the time varying randomization and provide better transition from exploration to exploitation. α minimizes with each iteration and is expressed as:

$$\alpha = C_j \times \exp\left(\frac{-t}{t_m}\right) \quad (13)$$

Where C_j is constant and t_m is maximum iterations.

Overcome form local optima*:* This phase helps to reduce the possibility of algorithm getting trapped in local optima by varying the searching direction.

$$F = \begin{cases} 1 \; when \; rand_3 \leq 0.5 \\ -1 \; otherwise \end{cases} \quad (14)$$

where $rand_3$ random number and it ranges from 0 to 1.

The digging phase of honey badger is updated using (15)

$$y_{new} = y_{prey} + F \times \alpha \times I \times y_{prey} + F \times \alpha \times rand_4 \times D_j \times |\cos(2\pi rand_5) \times [1 - \cos(2\pi rand_6)]| \quad (15)$$

where α is capacity of HB for getting food, $rand_4$, $rand_5$ and $rand_6$, are the various random numbers and it ranges from 0 to 1.

The HBA algorithm evaluates fitness in each iteration based on the objective function and update the HB population based on digging phase. As the HBA algorithm has the drawback of getting trapped in local optima and poor convergence [5] the results obtained in the digging phase is further optimized by passing to the second algorithm, FSA. The FSA algorithm is a global exploration algorithm which improves solution with reduced randomness in input sample and hence iterate over the obtained input to generate final solution.

FSA algorithm is designed to imitate the foraging and migrating behavior of gregarious birds, flamingos. The flamingos are migratory birds which search for food as a group and sing to each other to communicate the availability of food source. This is represented as the foraging characteristics of flamingos and is coded as foraging phase. The flamingos migrate to different location if food is scarce in a particular region and cannot feed the existing population. This behavior is modelled as migration phase. The FSA is proved to have faster convergence rate and

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7260
***Article Received**: 30 March 2023 **Revised**: 28 April 2023 **Accepted**: 14 June 2023*

accuracy compared to different algorithms such as Ant Colony optimization (ACO), Particle Swarm Optimization (PSO), Grey Wolf optimizer (GWO) [4] etc. The mathematical model for FSA is detailed below:

To characterize the foraging behavior of flamingos let us assume that search agent with more food availability in k^{th} dimension is yb_k and the location of j^{th} flamingo in k^{th} dimension of the flamingo population is $y_{jk.}$ The maximum distance that can be covered by flamingo's beak is represented as $J_2 \times |J_1 \times yb_k + \gamma_2 \times y_{jk}|$, where J_1 is a random number following normal random distribution, γ_2 is random number between -1 and +1and J_2 is a random number. While foraging for food, the bipedal movement of flamingo is represented as γ_1 *x* yb_k, where γ_1 is random number and it ranges from -1 or +1. The sum of distance covered by the beak and claws of flamingos can be represented as (16).

$$C_{jk}^{t} = (\gamma_1 + yb_k) + (J_2 \times |J_1 \times yb_k^t + \gamma_2 \times y_{jk}^t|) \quad (16)$$

The initial and updated locations of flamingos in foraging stage is represented in (17) and (18) respectively:

$$y_{jk}^{t}(0) = \frac{R_{jk}}{R_{avg}} \quad (17)$$

$$y_{jk}^{t+1} = \frac{(y_{jk}^t + \gamma_1 + yb_k^t + J_2 \times | J_1 \times yb_k^t + \gamma_2 \times y_{jk}^t |)}{I} \quad (18)$$

where, j^{th} flamingo in k^{th} dimension during $(t+1)^{th}$ and t^{th} iterations are represented as $y_{jk}^{t+1}$and y_{jk}^{t}. I is the diffusion factor.

The migration characteristics of flamingos is represented as:

$$y_{jk}^{t+1} = y_{jk}^{t} + \alpha \times (yb_k^t - y_{jk}^t) \quad (19)$$

where α is Gaussian random number

$$\eta_{jk}(t) = \frac{load_k}{R_{jk}} \quad (20)$$

Here, when R_{jk} increases, $\eta_{jk}^{(t)}$ and $load_k$ are decreased.

The migratory flamingos are selected based on evaluating the fitness of flamingos. In our problem, the tasks are arriving in time constrained manner which has deadlines for completion. Each fog node has a maximum capacity expressed as duration time. The fog node accepts only the tasks whose deadline can be satisfied within the duration time allotted for it. So, the migration phase of flamingos is modified to consider the deadline of tasks along with the fitness value.

The complete workflow of HB-FS can be modelled as given below:

Algorithm 1. Pseudo code of proposed HB-FS Algorithm

1. Set parameters like C_j, β, *M, no. of flamingos, number of nodes, Max* and t_m.
2. Initialize the population of Flamingos
   2.1. Initialize the honey badger population randomly
   2.2. Evaluate fitness of each honey badger and save the best position
   2.3. For *t= 1* to t_m, do
   2.4. Update the density factor, α
   2.5. For *j=1 to M* do
   2.6. Compute smell intensity of prey, I_n
   2.7. Perform position update using digging phase and compute fitness in each iteration.
   2.8. End for loop
   2.9. End for loop
   2.10. Return intermediate result
3. Initialize the foraging and migrating characteristics of flamingos
4. For i=1 to *Max i* , do
5. Compute foraging characteristics using Eq. (18) and
6. Compute migration characteristics using Eq. (19)
7. Evaluate fitness and update flamingo location
8. Update the best solution
9. End for loop
10. Return the final optimized solution.

IV. RESULTS AND DISCUSSION

The proposed methodology is tested against different existing works to evaluate the performance. The simulations are done using the iFogSim simulator toolkit which is Java based. The implementation uses synthetically generated IoT online gaming dataset for evaluation purpose. The processor used is AMD Ryzen 5, 3500U with Radeon Vega Mobile @ 2.10 GHz. The RAM memory used is 8GB with 64-bit operating system. The different parameter value settings used for the simulation is shown in Table 1. below:

TABLE 1. PARAMETER CONFIGURATIONS

Parameters	Value
Sensor nodes	2500
No. of fog nodes	10-25
Max no of iterations	50
B_l	700 Mb/s
B_c	37 Mb/s
L_{jk}	2-20 ms
L_{jc}	20-30 ms

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7260
***Article Received**: 30 March 2023 **Revised**: 28 April 2023 **Accepted**: 14 June 2023*

A. Performance Evaluation

The proposed methodology is tested against different algorithms such as Genetic algorithm (GA), Slap swarm algorithm (SSA), and GWO (Grey wolf optimization), Bee life algorithm (BLA) [19], modified particle swarm optimization (MPSO) [20], Smart Ant Colony Optimization (SACO [15] , Throttle and RR (round robin). The major evaluation parameters used are latency, average response time, standard deviation (SD), execution time, degree of imbalance *(DI)* etc. The results of the comparison are discussed in section below:

(a) Scenario 1 & 2

Initially, the evaluation is carried out using 10 fog nodes. The parameters used for the evaluation are λ_{ij}=15,10,5 and μ_{ij}=50,75,100. The proposed model is compared with different algorithms for evaluating the latency time. Evaluations show that proposed model offers lesser latency than the compared methods as shown in Fig 2(a) below. The evaluations also shows that latency time is increased when we increase the number of sensors. When the number of sensors is 2500, the latency time of RR is 126s, Throttle is 104s, MPSO is 102s, BLA is 101s, SACO is 97s, SSA is 100s, GA is 116s, GWO is 99s and the proposed model is 80s respectively.

For scenario 2, the parameters chosen are λ_{ij}=50,40,30 keeping the service rate and number of fog nodes same as scenario 1. The observations shows that latency can be reduced if the data rate is increased. When the number of sensors is 1750, the latency of RR is 107s, Throttle is 98s, MPSO is 97s, BLA is 96s, SACO is 94s, SSA is 77s, GA is 78s, GWO is 77s and the proposed model is 60s respectively.

(b) Third and fourth scenario

The third scenario is modelled by increasing the number of fog nodes to 25. The other parameters used are λ_{ij}=50,40,30 and μ_{jk}=400,350,300. The figure 3(a) shows that there is slight change in the latency when service rate is increased. When the number of IoT sensors are 2000, latency time of RR is 116s, Throttle is 114s, MPSO is 114s, BLA is 112s, SACO is 110s, GWO is 100s, GA is 121 s and SSA is 105s and the proposed model is 67s respectively.

Figure 3(b) shows the outcomes of the fourth scenario. Here, 25 sensors are considered, λ_{ij}=30,25,20 and μ_{jk}=400,350,300. It is observed that number of IoT sensors are 1750, latency time of RR is 110 s, Throttle is 107s, MPSO is 107s, BLA is 106s, SACO is 101s, GWO is 120s, GA is 123s and SSA is 122s and the proposed model is 64s. From all the four scenarios it can be proved that the proposed model generated lesser latency under different conditions.

Figure 4 represents the response time comparison of the proposed and existing algorithms like RR, ACO, PSO GWO, GA and SSA. The average response time for different tasks are observed on every iteration. In this case, the ART is maintained consistently even as the total tasks are increased by providing many IoT sensors. When compared to the existing models, proposed optimization maintains less response time when the IoT sensors are increased.

The execution time comparison of different algorithms is presented in Figure 5. The results show that for offloading process, the average time taken by the approaches like SACO, MPSO, throttle, RR, BLA, GWO, GA SSA and proposed approach are 107ms, 113.2ms, 115ms, 122ms, 110ms, 165.4ms, 81ms, 76ms and 45ms respectively. The proposed model converges with lesser execution time.

Degree of imbalance is another evaluation measure which evaluate how well the algorithm can balance the available load It is computed using (21).

$$DI = \frac{Max(R_k) - Min(R_k)}{R_{avg}} \quad k = 1,2, \ldots . N_{node} \tag{13}$$

Figure 6 depicts the comparison of degree of imbalance of proposed algorithm with existing algorithms. The results shows that the proposed algorithm maintains better degree of imbalance under different load conditions and can be effectively used for balancing the load. When the sensors are increased, the degree of imbalance is maintained in a balanced manner.

Standard deviation (SD): It is used for evaluating the load distribution between fog nodes. For smaller ranges, largely balanced nodes are obtained. It is computed as:

$$SD = \frac{\sqrt{\sum_k (R_k - R_{avg})^2}}{N_{node}} \tag{22}$$

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7260
***Article Received**: 30 March 2023 **Revised**: 28 April 2023 **Accepted**: 14 June 2023*

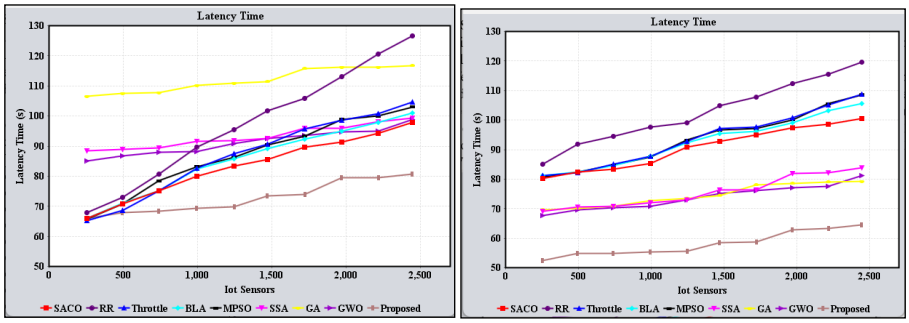

Fig. 2. Latency time comparison for (a) scenario 1 and (b) scenario 2

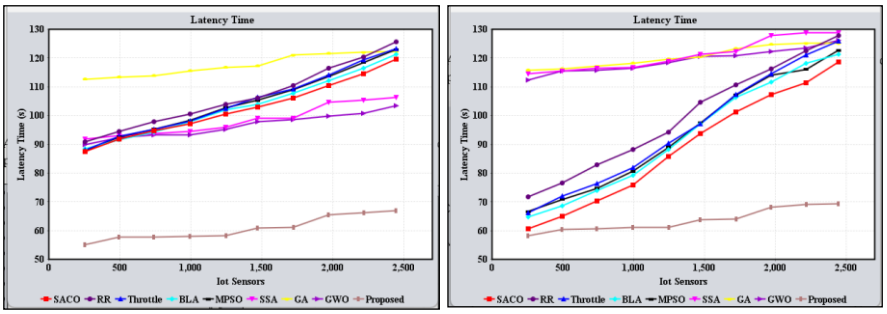

Fig. 3. Latency time comparison for (a) scenario 3 and (b) scenario 4

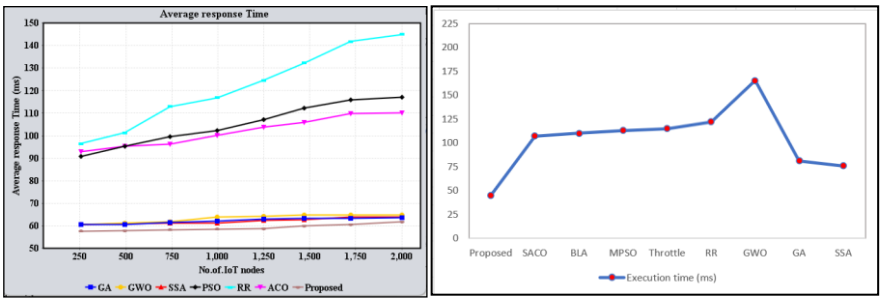

Fig. 4. and 5. Average response time and execution time comparison of the proposed and existing algorithms.

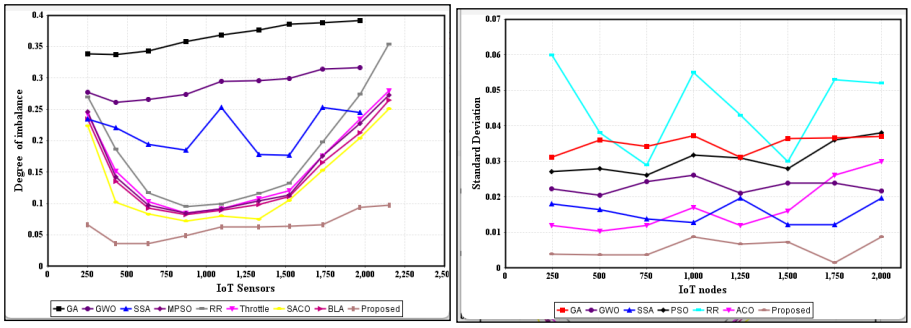

Fig. 6. and 7. Degree of imbalance and standard deviation comparison

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7260
***Article Received**: 30 March 2023 **Revised**: 28 April 2023 **Accepted**: 14 June 2023*

is the difference on response time of entire offloaded tasks from ART. From the figure it is clearly proved that the proposed task offloading technique performs better when compared to RR, ACO, PSO, GWO, GA and SSA respectively.

From the experimental analysis, it is noted that the proposed model achieves better latency time, average response time, standard deviation, execution time and degree of imbalance compared to the existing techniques. The observations from the analysis shows that by optimizing these parameters the QoS can be enhanced. Thus, the proposed model can be efficiently used for task offloading in latency sensitive IoT applications.

V. CONCLUSION

The tremendous growth of IoT applications over the past few years has increased the dependence of technology on cloud resources. The application generates millions of data every second. Cloud computing offers promising solutions to IoT applications for data storage, processing and analytics. However, latency sensitive IoT applications finds it difficult to completely depend on cloud for processing as the delay induced by remote cloud is intolerable. The introduction of fog computing offers timely delivery of services to IoT applications increasing the popularity of IoT-Fog-Cloud model. The task offloading in Fog layer is crucial as it impacts the QoS and performance of connected applications. This work focuses on designing an effective offloading approach using the combination of two novel meta-heuristic algorithms, Honey Badger Algorithm and Flamingo Search Algorithm. The proposed HB-FS algorithm utilizes the digging phase of HBA to initialize the population for FSA algorithm. The optimization function is designed to optimize different parameters like network latency, response time, resource utilization etc. The performance evaluations are carried out under different scenarios to prove the efficacy of proposed model. The quantitative results depicts that the proposed work performs better than compared techniques in terms of network latency, average response time, execution time. The model also offers better load balancing making it suitable for applications with varying load conditions. In future, this work will be considering the scalability issues and security factors to enhance the QoS and performance further.

REFERENCES

[1] Z. Qu, Y. Wang, L. Sun, D. Peng, and Z. Li, "Study QoS optimization and energy saving techniques in cloud, Fog, EDge, and IoT," Complexity, vol. 2020, 2020, doi: 10.1155/2020/8964165.

[2] S. Aljanabi and A. Chalechale, "Improving IoT Services Using a Hybrid Fog-Cloud Offloading," IEEE Access, vol. 9, pp. 13775–13788, 2021, doi: 10.1109/ACCESS.2021.3052458.

[3] F. A. Hashim, E. H. Houssein, K. Hussain, M. S. Mabrouk, and W. Al-Atabany, "Honey Badger Algorithm: New metaheuristic algorithm for solving optimization problems," Math. Comput. Simul., vol. 192, pp. 84–110, 2022, doi: 10.1016/j.matcom.2021.08.013.

[4] "Flamingo Search Algorithm: A New Swarm Intelligence Optimization Algorithm | Enhanced Reader." .

[5] A. M. Nassef, E. H. Houssein, B. E. din Helmy, and H. Rezk, "Modified honey badger algorithm based global MPPT for triple-junction solar photovoltaic system under partial shading condition and global optimization," Energy, vol. 254, p. 124363, Sep. 2022, doi: 10.1016/J.ENERGY.2022.124363.

[6] M. Yu, A. Liu, N. N. Xiong, and T. Wang, "An Intelligent Game-Based Offloading Scheme for Maximizing Benefits of IoT-Edge-Cloud Ecosystems," IEEE Internet Things J., vol. 9, no. 8, pp. 5600–5616, 2022, doi: 10.1109/JIOT.2020.3039828.

[7] R. O. Aburukba, M. AliKarrar, T. Landolsi, and K. El-Fakih, "Scheduling Internet of Things requests to minimize latency in hybrid Fog–Cloud computing," Futur. Gener. Comput. Syst., vol. 111, pp. 539–551, 2020, doi: 10.1016/j.future.2019.09.039.

[8] J. Y. Baek, G. Kaddoum, S. Garg, K. Kaur, and V. Gravel, "Managing Fog Networks using Reinforcement Learning Based Load Balancing Algorithm," IEEE Wirel. Commun. Netw. Conf. WCNC, vol. 2019-April, pp. 1–7, 2019, doi: 10.1109/WCNC.2019.8885745.

[9] A. A. Alli and M. M. Alam, "SecOFF-FCIoT: Machine learning based secure offloading in Fog-Cloud of things for smart city applications," Internet of Things, vol. 7, no. 2019, p. 100070, 2019, doi: 10.1016/j.iot.2019.100070.

[10] M. Abbasi, E. Mohammadi Pasand, and M. R. Khosravi, "Workload Allocation in IoT-Fog-Cloud Architecture Using a Multi-Objective Genetic Algorithm," J. Grid Comput., vol. 18, no. 1, pp. 43–56, 2020, doi: 10.1007/s10723-020-09507-1.

[11] M. I. Bala and M. A. Chishti, "Offloading in cloud and fog hybrid infrastructure using iFogSim," Proc. Conflu. 2020 - 10th Int. Conf. Cloud Comput. Data Sci. Eng., pp. 421–426, 2020, doi: 10.1109/Confluence47617.2020.9057799.

[12] X. Liu, J. Yu, Z. Feng, and Y. Gao, "Multi-agent reinforcement learning for resource allocation in IoT networks with edge computing," China Commun., vol. 17, no. 9, pp. 220–236, 2020, doi: 10.23919/JCC.2020.09.017.

[13] V. Meena, M. Gorripatti, and T. Suriya Praba, "Trust Enforced Computational Offloading for Health Care Applications in Fog Computing," Wirel. Pers. Commun., vol. 119, no. 2, pp. 1369–1386, 2021, doi: 10.1007/s11277-021-08285-7.

[14] M. Keshavarznejad, M. H. Rezvani, and S. Adabi, "Delay-aware optimization of energy consumption for task offloading in fog environments using metaheuristic algorithms," Cluster Comput., vol. 24, no. 3, pp. 1825–1853, 2021, doi: 10.1007/s10586-020-03230-y.

[15] A. Kishor and C. Chakarbarty, "Task Offloading in Fog Computing for Using Smart Ant Colony Optimization," Wirel. Pers. Commun., no. 0123456789, 2021, doi: 10.1007/s11277-021-08714-7.

[16] P. Kaur and S. Mehta, "Improvement of Task Offloading for Latency Sensitive Tasks in Fog Environment," Lect. Notes Data Eng. Commun. Technol., vol. 74, pp. 49–63, 2022, doi: 10.1007/978-981-16-3448-2_3.

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 11 Issue: 6
DOI: https://doi.org/10.17762/ijritcc.v11i6.7260
Article Received: 30 March 2023 Revised: 28 April 2023 Accepted: 14 June 2023

[17] A. Robles-Enciso and A. F. Skarmeta, "A multi-layer guided reinforcement learning-based tasks offloading in edge computing," Comput. Networks, vol. 220, no. September 2022, p. 109476, 2023, doi: 10.1016/j.comnet.2022.109476.

[18] Yakubu, I.Z., Murali, M. An efficient meta-heuristic resource allocation with load balancing in IoT-Fog-cloud computing environment. J Ambient Intell Human Comput (2023). https://doi.org/10.1007/s12652-023-04544-6

[19] S. Bitam, S. Zeadally, and A. Mellouk, "Fog computing job scheduling optimization based on bees swarm," Enterp. Inf. Syst., vol. 12, no. 4, pp. 373–397, 2018, doi: 10.1080/17517575.2017.1304579.

[20] A. Al-Maamari and F. A. Omara, "Task scheduling using PSO algorithm in cloud computing environments," Int. J. Grid Distrib. Comput., vol. 8, no. 5, pp. 245–256, 2015, doi: 10.14257/ijgdc.2015.8.5.24.