International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

Enhancing Resource Utilization and Performance in IoT with Fog Computing: A Novel Data Migration Approach Using SCCSO and SCPSO

Dipti Prava Sahu[*1], Biswajit Tripathy[2], Leena Samantaray[3]

[1]Research Scholar, Department of Computer Science & Engineering, Biju Patnaik University of Technology, Rourkela, Odisha, 769004, India

[2]Professor, Master of Computer Applications, Einstein College of Computer Application and Management, Khurda, Odisha, 752060, India

[3]Professor & Principal, Department of Electronics and Communication Engineering, Ajay Binay Institute of Technology, Cuttack, Odisha, 753014, India

E-mail: diptiparva29@gmail.com[1], biswajit69@gmail.com[2], leena_sam@rediffmail.com[3]

Corresponding Author : Dipti Prava Sahu

Abstract

Recently, the rapid growth of data in Internet of Things (IoT) environments has led to significant advancements in fog and mobile edge computing. However, these technologies face challenges in terms of inefficient scheduling, which can result in larger delays compared to traditional cloud computing. Moreover, security and privacy concerns arise due to the diverse range of services offered in the consumption areas of IoT devices. To address these issues, a data migration procedure is proposed in the fog computing paradigm to optimize metrics such as latency, response time, and effective resource utilization. In this approach, Sequence Cover Cat Swarm Optimization (SCCSO) and Sequence Cover Particle Swarm Optimization (SCPSO) are employed in the data migration process to efficiently allocate resources in the fog environment. The main goal is to minimize the replication and integration of data present in the cloud communication storage environment. Using these optimization protocols, the data migration process aims to achieve better performance compared to other scheduling algorithms. To evaluate the effectiveness of the proposed approach, extensive testing is conducted in the iFogsim environment. The results demonstrate that the SCCSO and SCPSO protocols outperform other scheduling algorithms in terms of energy usage, execution time, and average response time. This implies that the data migration procedure contributes to improved resource utilization and overall system performance in fog computing scenarios.

Keywords: Internet of Things, Fog computing, Mobile edge computing, Data migration, Scheduling algorithms, Latency, Response time, Resource utilization, Security, Privacy.

1. Introduction

Fog computing is a decentralized network computing approach that brings memory and computational resources closer to the endpoints, utilizing units among datacenters and IoT devices in various architectures [1]. It employs networking tools such as gateways, exchanges, configuration packages, ground stations, and tunnels, each with dedicated networking, storage, and compute services. The term "fog computing" was first coined by Linksys to address the limitations of cloud computing [2]. Fog computing has become a powerful solution in industries like healthcare, smart glasses, and entertainment. It works in conjunction with cloud environment components to reduce processing times, speed up computations, and lower costs [3]. There are two types of edge devices: resource-rich cloudlets, which act as covered cloud data centers and offer mobile devices substantial processing power with low latency, and energy-efficient devices like access points, preconfigured boxes, and base stations. The growing interconnection of IoT-based smart solutions [4], such as home automation, sustainable cities, transportation, monitoring devices, and wearable computing, has captured the interest of academia and businesses. However, the

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

increasing number of connected devices also leads to higher power consumption and performance degradation. Virtualization is less effective for IoT due to latency concerns [5], making energy- and performance-conscious computational resource services crucial. Resource management in IoT mobile ad-hoc networks poses challenges due to the dynamic nature of endpoint devices' bandwidth, storage, computation, and latency. To maintain a good level of service, effective resource management is necessary [6]. The MigCEP placement and migration approach has been recommended for resource management in both cloud and fog computing, considering reduced network utilization and edge latency constraints [7].

Figure 1 depicts the high-level architecture or conceptual representation of a Fog computing operation [8]. Fog computing is a decentralized computing paradigm that extends the cloud computing model by bringing computation, data storage, and networking closer to the edge of the network, nearer to the data sources and end-users. This proximity to the endpoints aims to reduce latency [9], bandwidth usage [10], and dependency on centralized data centers. The IoT Devices [11] are the endpoint devices, sensors, actuators, and other smart devices that generate data and interact with the environment. Examples could include smart home devices, wearables, industrial sensors, etc. Fog Nodes [12] are intermediate computing devices located closer to the IoT devices and act as the "fog" layer between the devices and the centralized cloud. Fog nodes process data locally and can provide faster responses to IoT device requests. Cloud Datacenters [13] are the traditional centralized data centers that provide cloud computing services. While Fog computing aims to perform computation at the edge, some tasks may still be offloaded to the cloud for more extensive processing or long-term storage. Gateways act as the communication bridges between IoT devices and the Fog nodes or cloud datacenters. Gateways [14] enable secure and efficient data transmission from devices to the Fog computing infrastructure. Networking Tools architecture may include various networking tools, such as exchanges, configuration packages, ground stations, and tunnels, to facilitate efficient data communication and management.

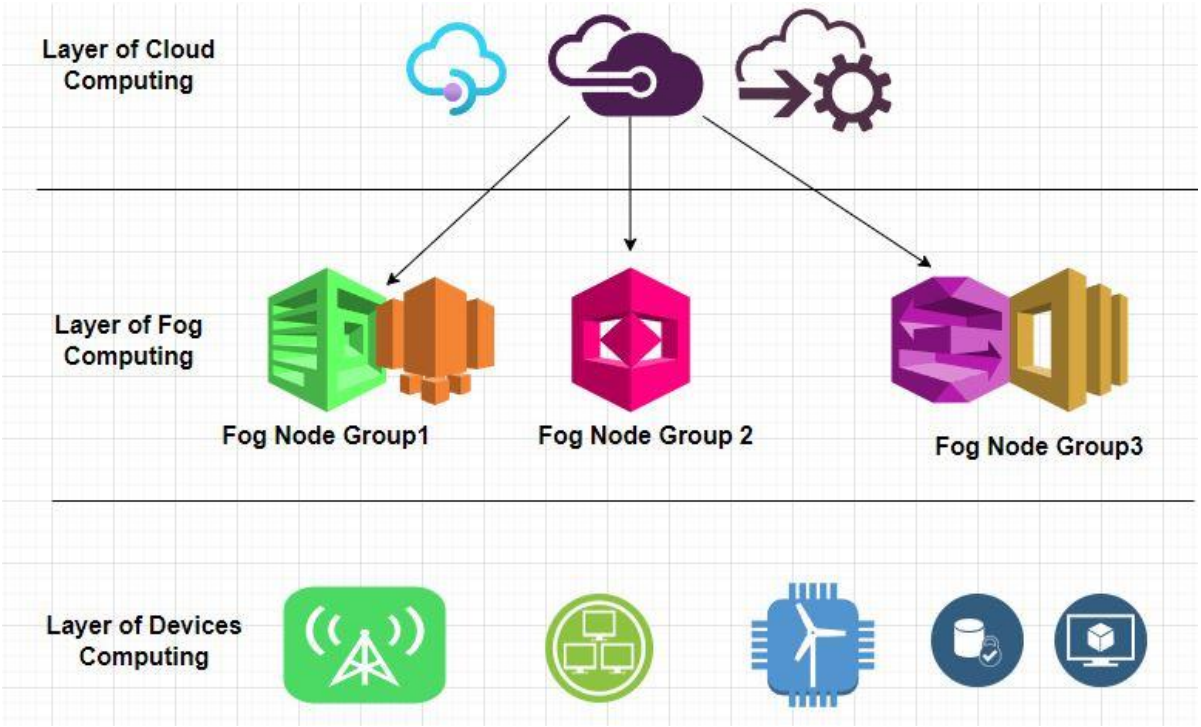

Fig.1. Architecture of the Fog computing operation

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal www.ijarst.in
ISSN: 2457-0362

Fog Services [15] are specifically designed to run at the Fog layer and provide computing, storage, and networking capabilities. Fog services help optimize performance and resource utilization for edge applications. Cloud Services are the traditional cloud-based services offered by cloud datacenters. In a Fog computing architecture [16], some tasks were offloaded to the cloud when local resources are insufficient or for more extensive data analysis. Edge Devices are the devices that exist at the network edge and serve as intermediary nodes between the IoT devices and the central Fog nodes or cloud datacenters The novel contributions of this work are as follows:

- Development of a data migration procedure for fog computing in IoT environments utilization of SCCSO and SCPSO in the data migration process.
- Optimization of metrics such as latency, response time, and effective resource use in the fog computing paradigm.
- Addressing security and privacy concerns related to diverse services in IoT device consumption areas.
- Efficient allocation of resources in the fog environment, minimizing replicated and integrated data in cloud communication storage.
- Outperforming other scheduling algorithms in terms of energy usage, execution time, and average response time.
- Improved resource utilization and overall system performance in fog computing scenarios.
- Promising solution for handling large IoT data and enhancing the efficiency and reliability of fog computing systems.

The subsequent sections of this paper will cover the system architecture, proposed system methodology, and problem formulation in Section 3, while Section 2 will discuss related work. The algorithms for the proposed method will be presented in Chapter 4. Section 4 will outline the quality characteristics and grading measures used for comparison, followed by the results and findings of the proposed technique in Section 4.

2. Related work

Using Big Data for Decentralized Management of IoT-Driven Healthcare Devices, [17] dispersed a bag of tasks to edge or fog nodes at the network's edge. These nodes are located at the network's periphery. The goal was to reduce the amount of time that mobile processes that operate on fog nodes take to execute as well as the amount of memory that they need. This evolutionary scheduling approach ensured that the trade-off between cost and execution time was preserved while scheduling jobs on fog and cloud resources in a manner that was appropriate for those tasks. The primary objective of the study was to identify a workaround for the scheduling problem that arises for programs that must do many activities within the context of a framework supporting dual virtualization. It was decided to apply the scheduling evolutionary approach that had been established by Xu, G, et al. [18], and the performance of this method was evaluated using a variety of work datasets on cloud and fog devices. The optimization criterion offered a compromise between the amount of time spent, the amount of money spent, and the satisfaction of the user. The virtualized layer, the customer layer, and the foggy layer make up the three tiers of the model that was proposed for the purpose of achieving optimal resource allocation in cloud-fog environments. When considering the total amount of time used, the network latency, and the cost of personal information, cloud resources were leveraged to fill up any gaps that may have occurred in the workload allocation between the client and the fog layer. The strategy, on the other hand, had the disadvantage of distributing resources before processing them and did not allow for the allocation of resources at run time.

The procedure that was proposed by Asheralieva, A, et al. [19] was divided into two parts: the distribution of jobs to virtual machines and the allocation and management of resources for the fog. Both the MPSO technique for job assignment and the MCSO approach for resource management were bio-inspired algorithms. MCSO was an acronym for "multi-objective particle swarm optimization." All elements of cloud-based applications were improved because of the strategy, including reliability, average response times, and resource use. Nevertheless, this concept

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal www.ijarst.in
ISSN: 2457-0362

could only be used in online or cloud-based settings. One contemporary use of fog computing settings was presented in [20], where a cost-effective solution was provided for controlling ground station access, workload split, and VM placement in the building of a medical cyber-physical system. This application was shown to be useful in managing ground station access, workload split, and VM placement. The issue was categorized as a mixed set of algebraic equations for ground station affiliation, job scheduling, and virtualization allocation. To find a solution to the problem, a heuristic strategy that was built on multiple linear programming was used.

In the research carried out by Lin, Y. et al. [21], several different scheduling strategies were used to improve the stability of the computing, scheduling, and operational processes of cloud systems. Ming and colleagues [22] developed a system to explore the problem of unequal power consumption as well as delays in the distribution of tasks in cloud-fog settings. To find the most effective solution to the issue of communication latency, the framework partitioned the problem into three subproblems that were each associated with one of the subsystems. After then, the Hungarian method was used to tackle these problems. According to the results of several simulations, environments with fog gave much reduced communication latency compared to cloud settings. On the other hand, the fact that the optimization was performed in a centralized method rather than a distributed one was one of the shortcomings of the system. Because of the system's complexity and the high expenses of connection and information exchange, implementing it under cloud cover architecture was a more difficult task than first anticipated.

In their study [23], Oueis et al. presented a solution to the problem of providing job scheduling services for cloud-based software systems. The strategy sought to reduce the complexity of the process while also decreasing the amount of power that was used while simultaneously meeting the demands and expectations of the users. The method consisted of two stages when it came to the distribution of the resources. During the first stage, resources were allotted to intelligent cells in accordance with a certain scheduling rule. During the second stage, clusters were developed to address any unmet requirements. Despite the many benefits, implementing this strategy inside a complex fog system proved to be a significant challenge. In Saravanakumar et al.'s [24] model for load balancing, components from LSTM theory and those related to fog were integrated. This was done to construct the model. By using a procedure known as cloud atomization, the method was able to transform virtual computers into a wide variety of physical nodes. To achieve this goal, it made use of clustered division and a predetermined number of resources. Furthermore, depending on the number of resources that were required, jobs or tasks were allocated across one or more VM nodes. Nevertheless, dynamic load balancing could not be accomplished using this model.

3. Problem Identification

The main problem addressed in this study is the need for sophisticated approaches to resource management in remote cloud computing systems, particularly in the context of task scheduling and resource utilization enhancement. The integration of two algorithms, SCPSO and SCSO, sets this research apart from previous studies to achieve two objectives simultaneously. The primary goal is to improve reaction times and resource utilization. The proposed system architecture comprises a cloudlet system with private and public cloud processing nodes. It consists of three levels: the consumer module, the planner, the haze devices (fog nodes), and the cloud data centers. The planner receives client requests and employs an iterative method to select the most suitable equipment for each job based on its CPU and memory requirements.

Figure 2 shows the fog computing framework system model. The SCPSO strategy is used to allocate tasks and determine the optimal solution for fault tolerance before employing the hybrid SCPSO-SCSO method. For evaluating the proposed approach, the iFogSim emulator is used to model the system. The study introduces a new nanoparticle heuristic model for strategic planning and task distribution. The SCPSO algorithm plays a key role in allocating responsibilities to fog devices based on task requirements and typical reflexes of cloud infrastructures. A hybrid approach, utilizing SCPSO and SCSO, is employed to manage fog devices with respect to CPU and memory requirements. As the user base of IoT grows, making the best use of resources becomes crucial for adequate service

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal www.ijarst.in
ISSN: 2457-0362

delivery. Dynamic resource allocation poses challenges, and this research aims to address this issue by building on the successful outcomes of MPSO (Modified Particle Swarm Optimization) for cloud-based systems. The solution for resource management is developed based on a modified version of MCSO (Modified Cat Swarm Optimization). The developed framework enables effective resource management and task load balancing in remote cloud computing systems. Tasks are planned to use SCPSO, considering user traffic and activities on relevant fog nodes and cloudlet. By selecting the most appropriate fog and cloud devices to handle requests, the proposed strategy enhances overall system performance. The developed framework enables effective resource management and task load balancing in remote cloud computing systems. Tasks are planned to use SCPSO, considering user traffic and activities on relevant fog nodes and cloudlet. By selecting the most appropriate fog and cloud devices to handle requests, the proposed strategy enhances overall system performance. The study focuses on finding the median system performance of cloud cover nodes following task scheduling.

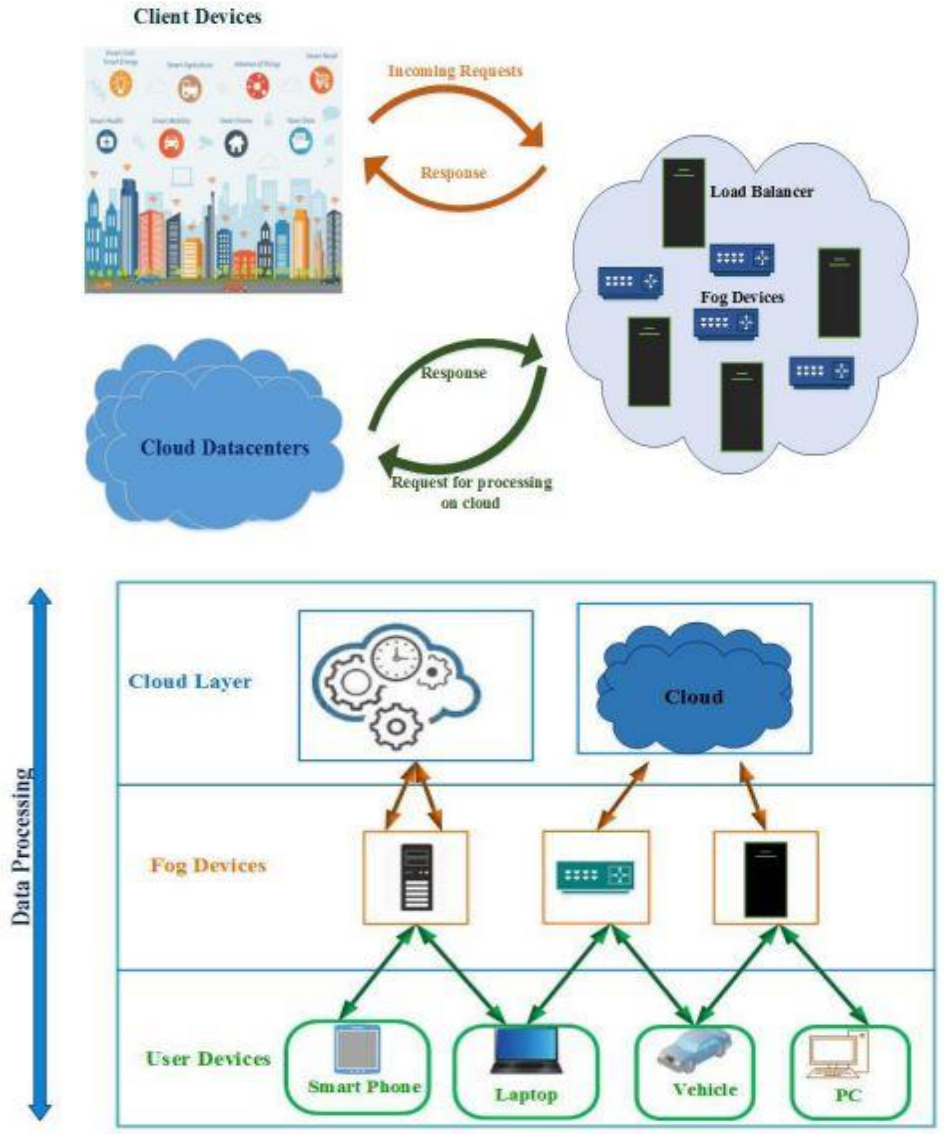

Fig.2.Fog Computing Framework System Model

4. Proposed Methodology

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal www.ijarst.in
ISSN: 2457-0362

4.1 SCA based task scheduling.

Once they have been grouped together, the fog fractals are able to communicate with one another using the communication component. The fog nodes assign the greatest priority to the work with the earliest deadline that also has the smallest total task size. Because the priority level of the tasks in this project could be adjusted and regulated in a dynamic manner, the order in which the operations were performed may shift as the project moved along. In earlier attempts, a scheduling system with a defined priority was suggested; however, its use in the task-carrying out process was not shown. If both jobs have the same hard deadline, then you should choose one of them at random. The scheduling of tasks at the fog node is described by Algorithm I.

4.2 ANN-based load balancing

This section provides functionality for load balancing using ANN. A back propagation learning technique is used in this system so that the workload is fairly distributed across all the fog nodes. The given method for controlling congestion is straightforward and efficient, and it can manage nodes sufficiently if there is sufficient training in the use of specific containers. An artificial network will evaluate the requirements of the market and contribute to the upkeep of user chores as required. Utilization of fog nodes is linked to increased levels of energy consumption as well as decreased levels of throughput. When compared to the loads carried by the other nodes in the fractal, the present load carried by one fog node seems to be excessive. If the load on a fog node reaches the limit that was defined, congestion control will be initiated. Within the scope of this study, we provide an adaptive routing limitation strategy that makes use of convolutional neural networks. An ANN, or artificial neural network, is a kind of supervised machine learning method that is used to calculate the current usage of the fog node. The ease with which this method is included into any kind of prediction software was the driving force behind the decision to disseminate it. Because of this, even if just a single fog node is underutilized or congested, this article guarantees that workloads are dispersed among many fog nodes so that the task is accomplished. Load balancing in fog computing is done primarily with the objective of reducing overall energy consumption. The three layers that make up our proposed architecture for an ANN are as follows: the input layer, the hidden layer, and the output layer. The designers assess the bandwidth needs of N fog nodes in the input layer as depicted in Equation (1).

$$(1+\mathrm{x})^{\mathrm{n}} = 1 + \frac{\mathrm{nx}}{1!} + \frac{\mathrm{n(n-1)x^2}}{2!} \quad (1)$$

At every fog node, the total number of tasks that are currently executing is S, the size of each task is $A(Ts)$, and the time it takes to simulate is T. The calculation of the average load for each fractal is done using Equation (2).

$$\mathrm{f(x)} = \mathrm{a_0} + \sum_{\mathrm{n=1}}^{\infty} \left(\mathrm{a_n} \cos\frac{\mathrm{n\pi x}}{\mathrm{L}} + \mathrm{b_n} \sin\frac{\mathrm{n\pi x}}{\mathrm{L}}\right) \quad (2)$$

The number of fog nodes in a fractal is denoted by the symbol N, the amount of time that has been left to complete the job that node j is working on is denoted by the symbol IRETj, and the amount of time that has been spent on the task is denoted by the symbol $ET(S)$. The activation function makes a prediction about the focus that will be placed on continuous throughout N fog devices, while the hidden layer determines a weight value according to the workload that is now being performed by the fog node.

$$[(\mathrm{x,y}),(\mathrm{a,b})] = \sqrt{(\mathrm{x-a})^2 + (\mathrm{y-b})^2} \quad (3)$$

Here, L is the symbol used to signify the constant multiplier. The load balancing threshold must be dynamic since the constant implementation of work that depends on EDF and the fog node translation of something like a task makes it impossible for static values to function well enough in the resource-constrained actual environment. Within the fractals of this work, network traffic is handled on an individual basis.

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal www.ijarst.in
ISSN: 2457-0362

Cloud data centers: If the fog fractals get corrupted while the job is being carried out, the cloud storage center will step in as the user's request administrator and supply the user with the apps that they have requested. When a cloud server is unable to fulfill a user's request, an alert is sent to the whole data center that manages the cloud. The job that was requested by the user is then finished using our innovative four-tier architecture.

Setup for the experiment: The proposed architecture for fog computing is developed, and then compared it to many other contexts for fog computing. The framework of our virtual world's simulation environment. For constructing the suggested architecture for fog computing, we decided to utilize iFogSim since it is built on top of the widely used CloudSim Simulator. We employed the Core CloudSim layer features, which oversee event handling, while we were working with the fog computing paradigm. To begin, we discussed the process by which the algorithm generates a fog device.

4.3 Sequence Cover Cat Swarm Optimization

SCCSO is a cutting-edge approach for system optimization that takes its cues from the behavior of cats as they pursue their prey. This behavior serves as the method's primary source of inspiration. The use of cat swarm optimization methods has shown to be effective in the resolution of several optimization issues, and these techniques provide several benefits over more traditional approaches. The optimization problem is based on the behavior of cats and comprises two submodels, "seeking mode" and "tracing mode," that each reflect a different aspect of cat behavior. The optimization problem is going to be made easier to solve if the SCCSO computation is successful in accomplishing its mission. The following is an outline of the stages involved in the approach based on cat swarms:

Step 1: It is an introduction to the optimization issue and parameters of the SCCSO algorithms: The power usage of the inter-range and battery systems is going to be lowered to bring the network's energy consumption down to a more manageable level. There is a total of N feline sensor networks, and these networks are aggregated one at a time into a high-dimensional feature optimal solution based on the feline sensor network that has the most accurate data. Using the MR (Coming from Different Sources Ratio), we can calculate the percentage of cats who engage in the seek-and-trace behavior. The linear combination that is described below is what is utilized to produce the problem and determine the value of the parameter. Due to the high level of activity seen at these areas, they have been given the "Xbest" designation.

Step 2: Seeking - the objective of the Spider Cats in the game If there are [N x (1-MR)] number of people who own cats, then the cats' mode should be changed to searching or tracing. Although they seem to be sound sleeping, the cats that make up this SCCSO submodel are really maintaining a vigilant watch on their environment to identify what step(s) they should take next. The most significant search criteria are as follows, SPC is used to search for ram groups, whereas CDC and SRD each set their change limits and recognition parameters at 0.2 and 0.04 kilobytes (kilobytes), respectively. SPC is used to find ram groups.

Step 3: Seeking stage: Make clones of cats using the SMP algorithm, saving the original cat's location in memory if the SPC condition is met. Be sure to receive the greatest deal for each of the most crucial variables by ensuring that you get the highest possible value. If none of the F are equal, then use Equation (4) to compute the probability of picking each coordinate point. Remove the rough sets and replace them with the best candidates possible.

$$Pi = (|f - fb|) / (fmax - fmin) \quad (4)$$

The ideal kind of training for modern fitness (f) and social interaction is found. If the objective is to decrease the volume of the solution.

$$fb = fmin \text{ otherwise } fb = fmax \quad (5)$$

Step 4: During the tracing phase of this interaction, the cats behave in a manner consistent with that of a beggar cat. Utilize Equation (5) to maintain accurate speed readings even when the cat is rotating. If the velocity is outside of

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal www.ijarst.in
ISSN: 2457-0362

the allowed range, the threshold point should be adjusted. It is recommended that Equation (6) be used to accurately record the current position of each kitten.

$$vod = vod + r* (xbest - xod) \quad (6)$$

$$xod = xod + vod \quad (7)$$

In this case, xod denotes the specific location of the cat. and $xbest$ is the best location of the cat in its current local environment; vod is a random value between 0 and 1 that reflects the cat's velocity in an M-dimensional space. where c is the acceleration constant, which ranges from 0.1 to 2.

Step 5: After you have reselected the cats and arranged them in a pattern that is either searching or tracing depending on the motion indicator, check to verify whether the well is still working before proceeding to the next step. If the response is "global greatest," then alter all the measures that are the most trustworthy; otherwise, return to step 2.

4.4 Sequence Cover Particle Swarm Optimization (SCPSO)

Simulation of flock social behavior is accomplished with the help of the PSO method, which is a population-based stochastic optimization model. Like eA in the sense that both consider the population as a whole and assign a health function to everyone. To add more complication to the situation, the arithmetic crossover operator of the EAs is analogous to the individual adjustment formula of the PSO, which isn't the only thing that's been influenced by simulations of social interaction. Another distinction is that in EAs, participants do not get any advantages because of their experiences. PSO has seen widespread usage as a solution for a broad range of optimization issues, including both discrete and continuous optimization issues, mostly due to its straightforward application. In PSO, a group of players known as particles will congregate in the region around the search zone. Each each particle stands for a one-of-a-kind answer to the optimization conundrum. The position of a particle is determined not only by its own best position but also by the experience of other neighboring particles. This experience refers to the location of the best particle in the surrounding area. The component of the swarm that holds the best position relative to its immediate surroundings is regarded as being the most advantageous overall. The finished algorithm is currently referred to by Gbest. In certain circles, this approach is referred to as the "best PSO for small neighborhoods." The fitness function adapts itself in response to the distance that separates the particles, and simultaneously, the optimization procedure and the global optimal value are computed.

The position of the particle currently represented as a_i, the current speed of the particle is b_i, The positions are ideal for a particle is c_i, the best place for the particle to be in its immediate environment. A particle's best position, also known as its personal best position, is the position that follows from it having the highest possible fitness value. Let's say that the objective function is denoted by f. After then, at the time step t, the best possible value for a particle is calculated using Equation (8):

$$\mathrm{db_i}(s+1) = \begin{cases} \mathrm{db_i}(s) \text{if } \mathrm{wg}\big(\mathrm{a_i}(s+1)\big) > \mathrm{dg}\big(\mathrm{b_i}(s)\big) \\ \mathrm{da_i}(s+1) \text{if } \mathrm{wg}(\mathrm{a_i}(s+1)) < \mathrm{dg}(\mathrm{b_i}(s)) \end{cases} \quad (8)$$

The whole swarm collaborated to determine which particle would be the most advantageous for the Gbest model. by picking the posture that is optimal for oneself personally. If the location of the best particle in the global average is represented by the vector according to Equation (9).

$$\check{\mathrm{b}}\{\mathrm{b_0}, \mathrm{b_1} \ldots . \mathrm{b_k}\} = \min\{\mathrm{g}\big(\mathrm{b_1}(s)\big), \ldots, (b_k(s))\} \quad (9)$$

In this case, the size of the swarm is denoted by the symbol k, and each dimension, $j \in 1, \ldots, Nd$ is listed after it. The step for updating the velocity has been given. As a result, the value $b_{i,j}$ denotes the j^{th} component of the velocity vector corresponding to the i^{th} particle. To finding the particle's velocity i, the equation (10) is utilized.

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

$$b_{i,j}(ws+1) = qb_{i,j}(ws) + c_1 m_{1,j}(ws) b_{i,j}(ws)) - b_{i,j}(s)) + c_2 m_{2,j}(s)(\widetilde{b(s)} - b_{i,j}(s)) \quad (10)$$

The weight of inertia is denoted by the symbol w, acceleration constants are denoted by the symbols c_1 and c_2, and the value of U is between 0 and 1. A scheduling cycle of one hundred milliseconds has been defined for the system configuration that we are experimenting with. The 64 fog nodes are currently being evaluated for their potential use in the process of allocating computing resources to users. Each of the 64 users sends a query to the cloud at the same time, taking between 10,000 and 20,000 kilobytes of data with them. The amount of delay experienced by each job ranges anywhere from 10 milliseconds to 100 milliseconds, depending on the quantity of data being processed. Figure 4 presents the iFogSim network architecture for your perusal. At the user base, we only established one fog data center in addition to the cloud datacenter that was already there (see Figure 5). If the virtualization layer does not have any additional activities, the fog will choose the data center that is geographically closest to it. In this section, we will begin by defining the parameters, and then proceed to carry out the mathematical calculations.

Response time: Response time is the amount of time that passes between when a user makes a request and when that request is received by the requesting interface. It is determined by adding the time it takes to process the request (t_r) and the time it takes for the behavior of the processing (t_{bp}).

$$rt = t_r + t_{bp} \quad (11)$$

Scheduling time: When it comes to the distribution of resources, scheduling time, also known as scheduling duration, is one of the most critical aspects. It details the beginning and ending timings of each activity using the notation $s_{te} + s_{et}$.

$$s_t = s_{te} + s_{et} \quad (12)$$

Load balancing rate: The distribution of the load across the fog nodes is evaluated so that congestion is reduced. It is possible to compute it using the current amount of work being done by each fog node.

Delay: The length of time that pass before the fog node's overall task is completed is referred to as the delay.

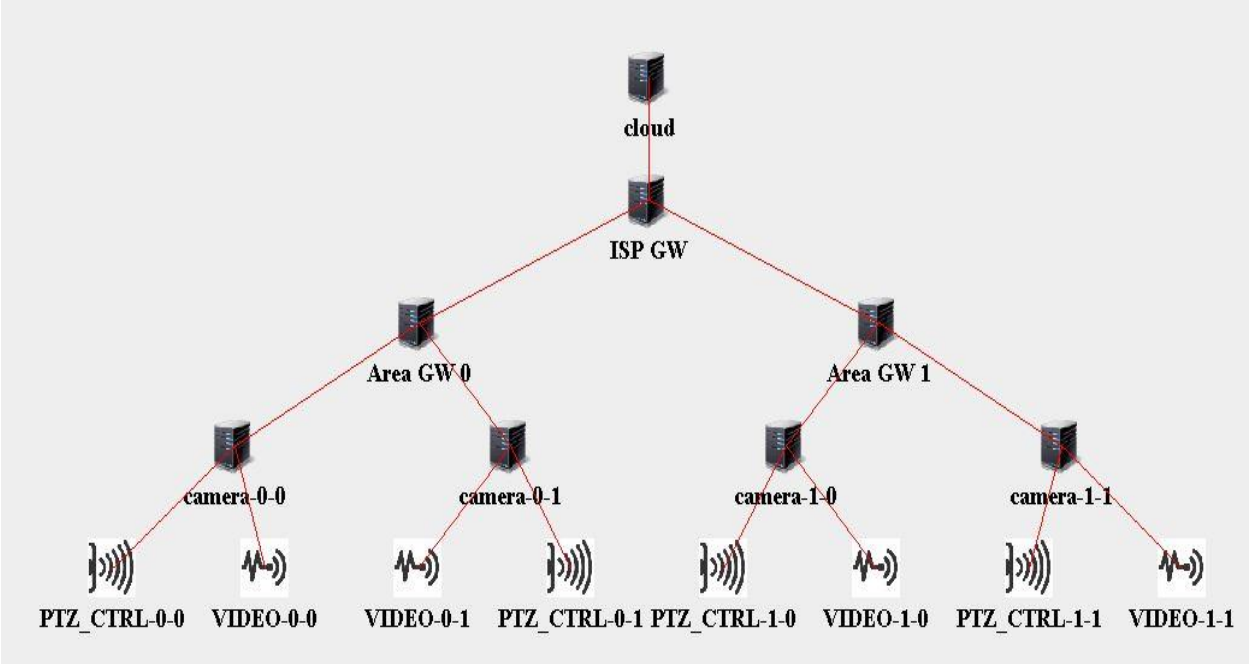

Fig.3.Deployment of Fog model classification

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

Energy consumption: It is a measure of the overall quantity of energy that is used by the whole system. Any of the system's components, such as sensors, fog nodes, and so on, may generally be used to monitor the system's overall energy usage. Kilojoules (KJ) are the units of measurement that are used to approximatively describe it.

Response time comparison: Response time is the amount of time that passes between when a user makes a request and when the first response to that request is sent. Response times are often faster in settings where the passage of time is variable. In Figure 6, a comparison is made between the suggested approach and existing methods such as BLA, simple sequencing, vertex division, Solidity, and MPGA, which are some of the techniques employed in BLA. However, the performance of the system in response to a user request is contingent on the latency and load of the fog nodes. The performance of the fog node is directly impacted by factors such as an imbalance in the load as well as an increase in the number of user requests. The BLA algorithm does not consider the load that is currently being carried by the fog node, which results in an inadequate identification and responsiveness of slopes. Should consider the headcount with five jobs assigned to each person. It takes 2500 milliseconds to finish each task, however the processing time decreases as the number of jobs increases. This work suggested an ANN-based fog architecture that effectively scheduled user requests at the fog layer. This architecture would be used for current usage prediction. The 1750 milliseconds are how long it takes us to finish all the jobs, which is 30 percent faster than data parallelism and % faster than BLA. When it comes to scheduling tasks, basic allocation, HEFT, and MPGA are all unsuccessful approaches.

Scheduling time contrast: The issue of work scheduling is a significant risk in fog computing, and it is imperative that an effective scheduling strategy be developed to mitigate this risk. Figure 7 presents a comparison of the amount of time that we advocate scheduling with the amount of time that is required by Simple sequencing, Solidity, and MPGA are all instances of CMaS. This experiment demonstrates how the amount of time spent preparing changes as there are more activities to choose from. It is possible to demonstrate the amount of time spent scheduling by the CMaS algorithm for a certain number of tasks, and we found that the suggested technique reduced the amount of time spent scheduling. CMaS is a system for scheduling tasks that is efficient and economical. The method that has been suggested will save 18.72 percent of the allotted time. This is since even when dealing with huge volumes of input data, the fog environment that we recommend has a higher processing network capacity.

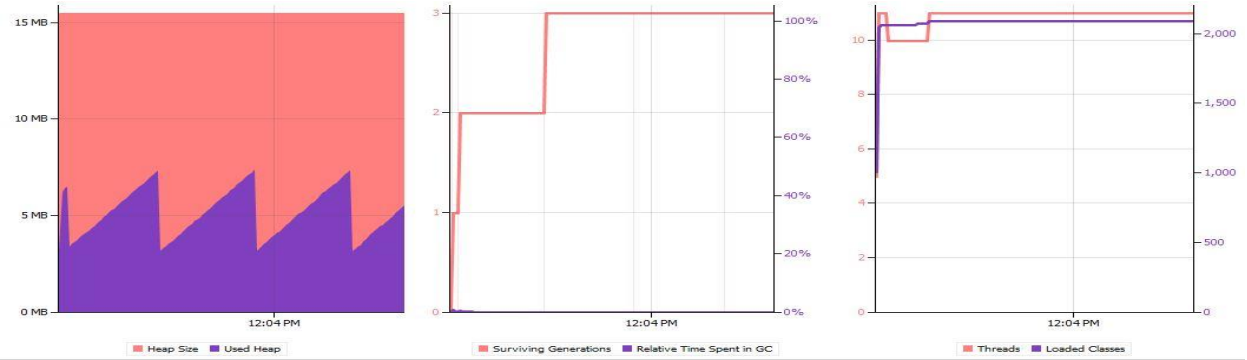

Fig.4.iFogsimResponse time Process

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

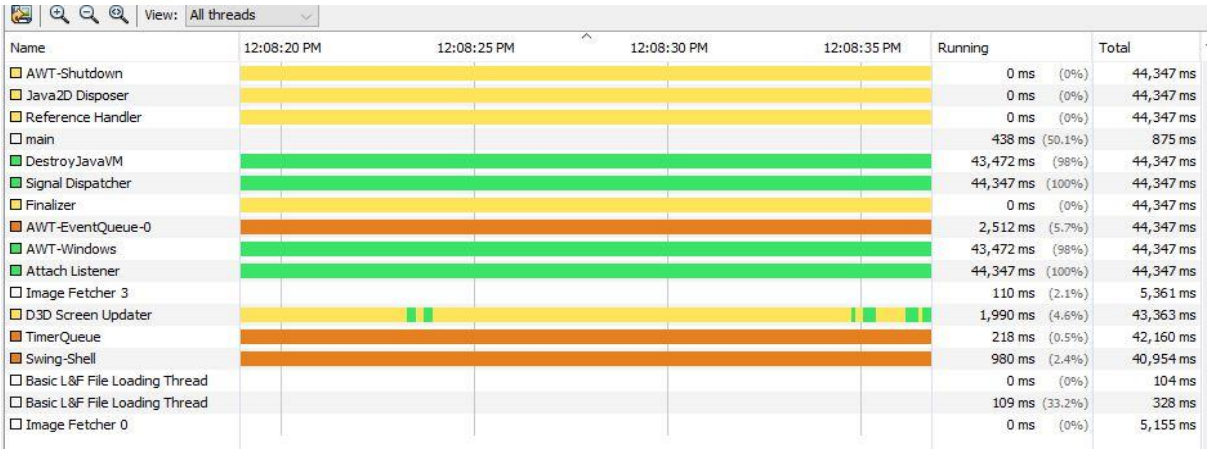

Fig.5.iFogsim scheduling time process.

Fault tolerant rate correlation: A fog node in a fog environment can only carry out a limited number of different tasks at any one time. When information on user tasks is merged in a single fog node, the pace at which load balancing is performed is slowed down. Figure 8 depicts the compute offloading rate that would occur in the recommended cloud environment while using DRAM. The purpose of this article is to distribute the burden that is placed on the network. The resource use of the fog virtualization technique is automatically kept in control if there is no user contact. On either side, the wide variety of deployment tasks is what is utilized to determine the load distribution across fog nodes. The amount of pressure increases if the target date for the completion of the work gets closer. Because of effectively managing the current load, we can arrange and carry out the tasks in an efficient manner. For example, five different activities are carried out by each fog node. A greater increase in latency was the effect of the previous Dynamic Resource Allocation Method. On the other hand, the fog design that we have provided provides an estimate of its utilization and furthermore incorporates a pressure check for each cloud level. If the operational liabilities of a fog network node exceed their cutoff threshold (L), then monitoring information is sent to the grain node. The grain node will then allocate the most recent client order to one of the neighboring fog layers.

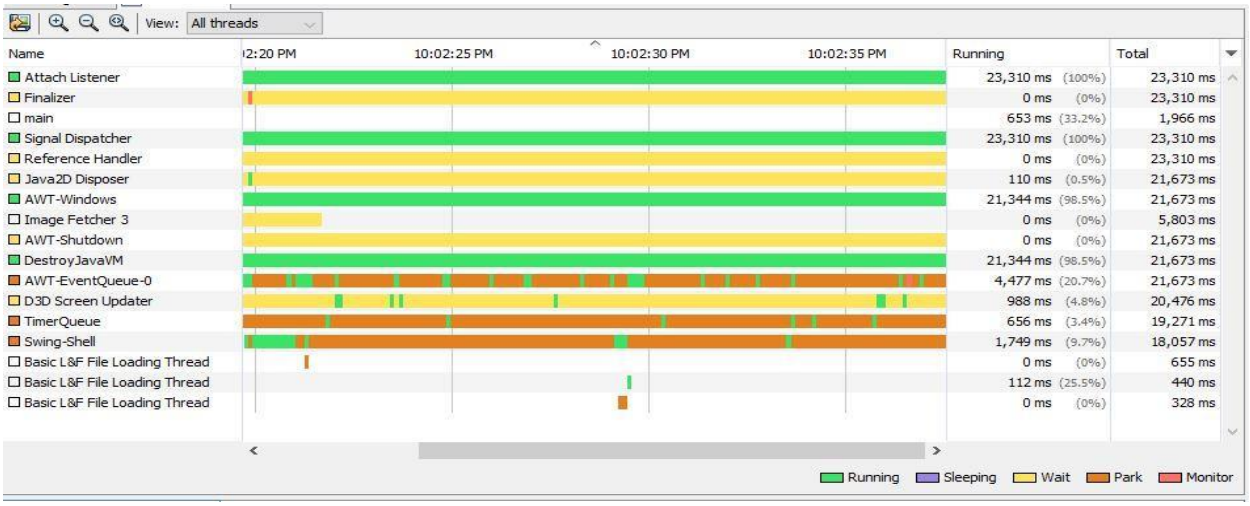

Fig.6.iFogsim scheduling task time Process.

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal www.ijarst.in
ISSN: 2457-0362

```
COORD MAP{}
Sensor type : EEG
Sensor type : EEG
Sensor type : EEG
Sensor type : EEG
Adding edge between PTZ_CTRL-0-0 & camera-0-0
Adding edge between PTZ_CTRL-0-1 & camera-0-1
Adding edge between Area GW 0 & ISP GW
Adding edge between Area GW 1 & ISP GW
Adding edge between VIDEO-0-1 & camera-0-1
Adding edge between PTZ_CTRL-1-0 & camera-1-0
Adding edge between VIDEO-1-1 & camera-1-1
Adding edge between VIDEO-0-0 & camera-0-0
Adding edge between VIDEO-1-0 & camera-1-0
Adding edge between camera-0-1 & Area GW 0
Adding edge between ISP GW & cloud
Adding edge between camera-0-0 & Area GW 0
Adding edge between camera-1-1 & Area GW 1
Adding edge between PTZ_CTRL-1-1 & camera-1-1
Adding edge between camera-1-0 & Area GW 1
##############################
{FogDevice [mips=1400 ram=1000 upBw=10000 downBw=270]=[Edge [dest=FogDevice [mips=2000 ram=4000 upBw=10000 downBw=10000]]]
##############################
sys:1261:651
```

Fig.7.Fognode device Classification process

Comparative delay plot as a result: The decrease of latency across the board should be one of the primaries focuses of fog computing, along with flexibility and stability. We investigated the delay that was present in the proposed framework for fog architecture. The decrease of latency across the board should be one of the primaries focuses of fog computing, along with scalability and dependability. We investigated the delay that was present in the proposed framework for fog architecture. The amount of work done by users ranges from 5 to 30 milliseconds, and the latency at the cloudlet in NSGA-II for five tasks is 2000 milliseconds; however, they suggest a much more sophisticated fog structure, specifically a Synthetic Fractal with an organized huge amount with each cloudlet, a limit of 1 Megabytes of space is available, which reduces the delay by 50% for five tasks, or 1000 milliseconds. NSGA-II is characterized by its simpler nature. The amount of time that must be delayed becomes much higher both as the number of tasks and the complexity of the tasks increase. The comparison latency is shown as a function of the total number of fog nodes in Figure 10. There are sixty-four fog nodes included in this simulation. The NSGAII algorithm makes effective use of the resources provided by fog nodes; nevertheless, this research did not consider current load monitoring or the possibility of fog node failure. This was the primary factor in the collapse of the system and the poor performance. Latencies, which are particularly high in NSGA-II, are shown to play a role by the graph to have an increasing impact as the number of fog nodes rises. The delay, on the other hand, is handled differently in our scenario.

Energy consumption comparative: To provide an accurate comparison of energy use, we considered the entire amount of time that was spent on task scheduling. Nevertheless, an effective decision strategy is necessary to cut down on energy use. We investigated the efficiency of energy use during job execution as well as task scheduling. Figure 9 shows the energy consumption values that were acquired using a range of different job numbers. On the other hand, the energy consumption of the fog node is much lower, however it is possible for it to increase if the fog node's duty is extended. Our intended fog environment needs 5500000KJ, while the DEBTS uses 11000000KJ (1.1104mW), and as a result, DEBTS uses more energy for fewer jobs than our environment does. This is since the architecture that we propose utilizes a novel strategy for the execution and scheduling of jobs. Because of this, the findings of our study indicate that the use of artificial fractals and present fog node usage should be exploited to increase performance. Because of this, we found that the performance of our suggested four-tier fog architecture was superior to that of earlier techniques in terms of response time (45% and 30%), scheduling time (18.72 percent) even

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal
www.ijarst.in
ISSN: 2457-0362

before comparing to CMaS, and load balancing rate (45% and 30%) only before comparing to BLA and Chart separation.

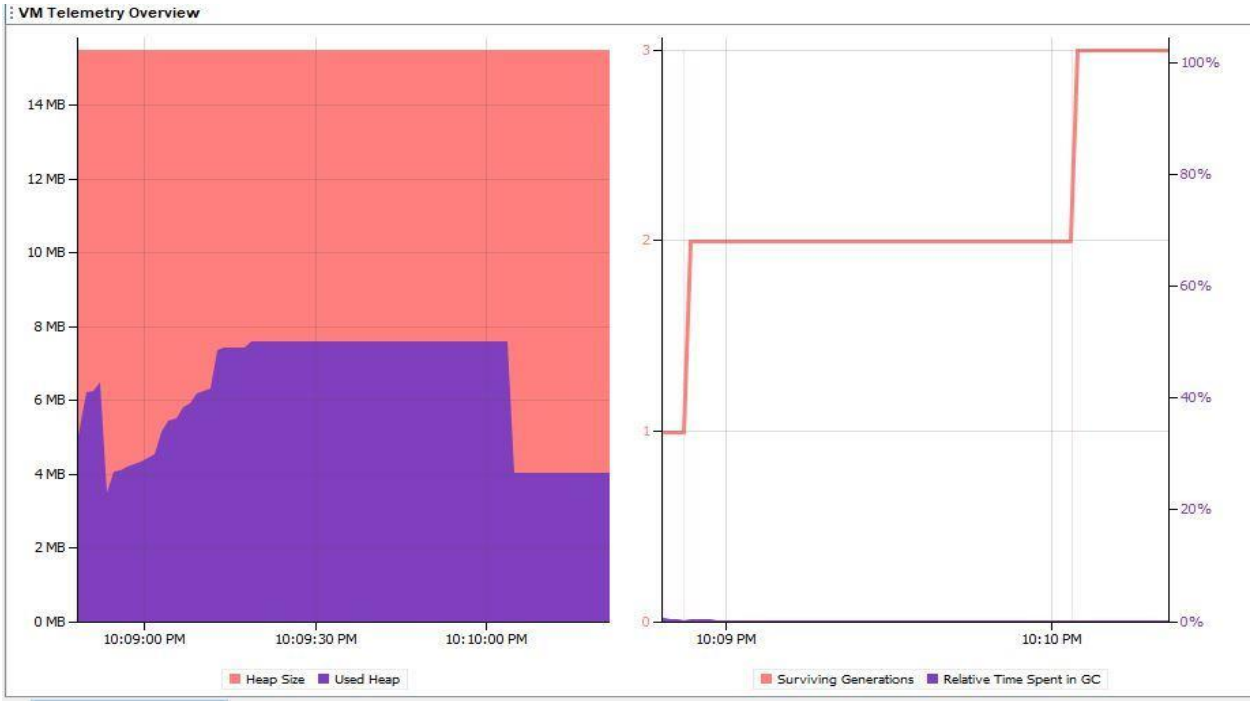

Fig.8. Load balancing rate of fog devices

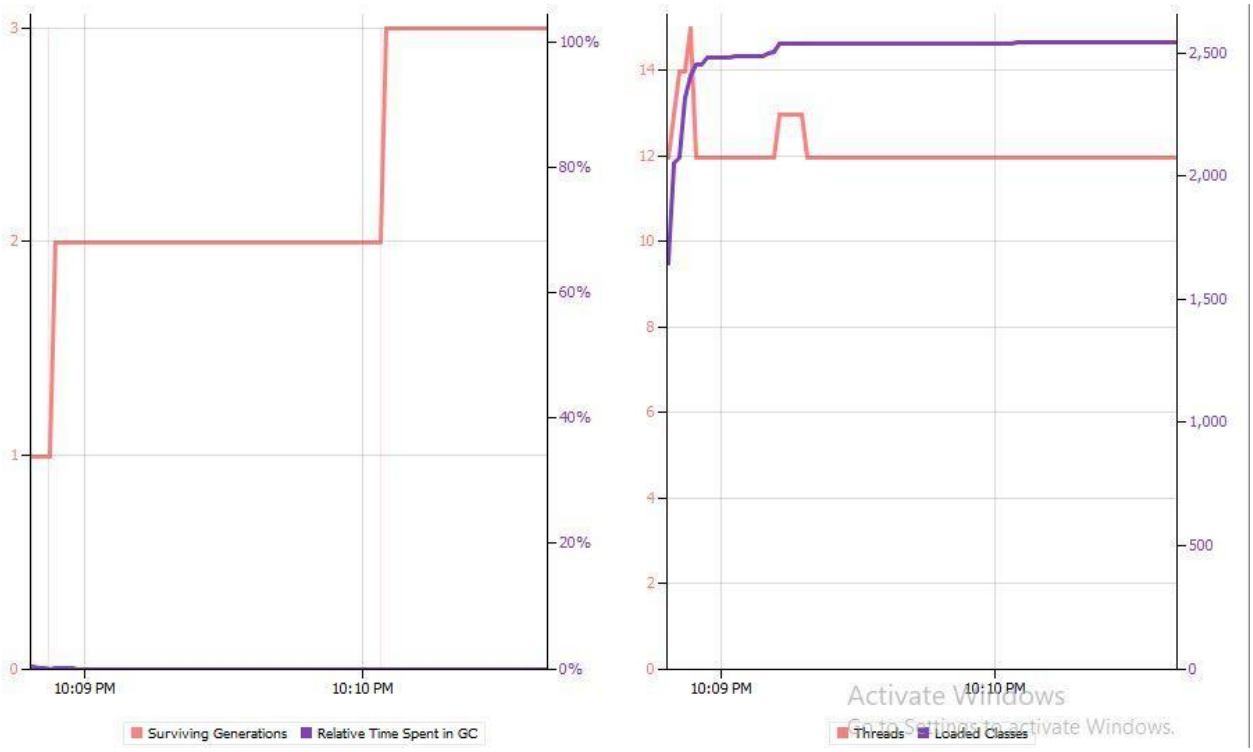

Fig.9. Delay of fog devices

6. Conclusion

Fog computing and mobile edge computing have emerged as essential technologies to handle the vast amounts of data generated in IoT environments. However, challenges related to inefficient scheduling, security, and privacy have been evident in these paradigms. To address these issues, a data migration procedure utilizing SCCSO and SCPSO has been proposed. The main objective of the data migration procedure is to optimize metrics such as latency, response time, and resource utilization in the fog computing paradigm. By minimizing data replication and integration in the cloud communication storage environment, the SCCSO and SCPSO protocols aim to enhance system performance. The evaluation of the proposed approach using the iFogsim environment has shown promising results. SCCSO and SCPSO outperformed other scheduling algorithms in terms of energy usage, execution time, and average response time. This suggests that the data migration procedure effectively improves resource allocation and overall system efficiency in fog computing scenarios.

References:

[1] Yin, Z., Xu, F., Li, Y., Fan, C., Zhang, F., Han, G., & Bi, Y. (2022). A Multi-Objective Task Scheduling Strategy for Intelligent Production Line Based on Cloud-Fog Computing. Sensors, 22(4), 1555.

[2] Abdel-Basset, M., Mohamed, R., &ELkomy, O. M. (2022). Knapsack Cipher-based metaheuristic optimization algorithms for cryptanalysis in blockchain-enabled internet of things systems. Ad Hoc Networks, 128, 102798.

[3] Sharma, V., & Tripathi, A. K. (2022). A systematic review of meta-heuristic algorithms in IoT based application. Array, 100164.

[4] Saravanan, T., &Saravanakumar, S. (2021, December). Privacy Preserving using Enhanced Shadow Honeypot technique for Data Retrieval in Cloud Computing. In 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N) (pp. 1151-1154). IEEE.

[5] Reddy, K. H. K., Luhach, A. K., Kumar, V. V., Pratihar, S., & Roy, D. S. (2022). Towards energy efficient Smart city services: A software defined resource management scheme for data centers. Sustainable Computing: Informatics and Systems, 100776.

[6] Ksouri, C., Jemili, I., Mosbah, M., &Belghith, A. (2022). Towards general Internet of Vehicles networking: routing protocols survey. Concurrency and Computation: Practice and Experience, 34(7), e5994.

[7] Singh, R. M., Awasthi, L. K., & Sikka, G. (2022). Towards Metaheuristic Scheduling Techniques in Cloud and Fog: An Extensive Taxonomic Review. ACM Computing Surveys (CSUR), 55(3), 1-43.

[8] Wadhwa, H., & Aron, R. (2022). A clustering-based optimization of resource utilization in fog computing. In Proceedings of International Conference on Advanced Computing Applications (pp. 343-353). Springer, Singapore.

[9] Saravanan, T., Saravanakumar, S., Rathinam, G., Narayanan, M., Poongothai, T., Patra, P. S. K., &Sengan, S. (2022). Malicious attack alleviation using improved time-based dimensional traffic pattern generation in UWSN. Journal of Theoretical and Applied Information Technology, 100(3).

[10] Almaiah, M. A., Hajjej, F., Ali, A., Pasha, M. F., &Almomani, O. (2022). A Novel Hybrid Trustworthy Decentralized Authentication and Data Preservation Model for Digital Healthcare IoT Based CPS. Sensors, 22(4), 1448.

[11] Alotaibi, M. (2022). Hybrid metaheuristic technique for optimal container resource allocation in cloud. Computer Communications.

[12] Liu, L., Khishe, M., Mohammadi, M., & Mohammed, A. H. (2022). Optimization of constraint engineering problems using robust universal learning chimp optimization. Advanced Engineering Informatics, 53, 101636.

[13] Jiang, Q., Zhou, X., Wang, R., Ding, W., Chu, Y., Tang, S., ... & Xu, X. (2022). Intelligent monitoring for infectious diseases with fuzzy systems and edge computing: A survey. Applied Soft Computing, 108835.

[14] Kumari, K. A., Sharma, A., Chakraborty, C., &Ananyaa, M. (2022). Preserving Health Care Data Security and Privacy Using Carmichael's Theorem-Based Homomorphic Encryption and Modified Enhanced Homomorphic Encryption Schemes in Edge Computing Systems. Big Data, 10(1), 1-17.

[15] D'Costa, D. R., & Abbas, D. (2022). 5G enabled Mobile Edge Computing security for Autonomous Vehicles. arXiv preprint arXiv:2202.00005.

[16] Liu, D., Zhang, Y., Jia, D., Zhang, Q., Zhao, X., & Rong, H. (2022). Toward secure distributed data storage with error locating in blockchain enabled edge computing. Computer Standards & Interfaces, 79, 103560.

[17] Saravanan, T., Ambikapathy, A., Faraz, A., & Singh, H. (2021). Blockchain and Big Data for Decentralized Management of IoT-Driven Healthcare Devices. In Convergence of Blockchain, AI, and IoT (pp. 57-81). CRC Press.

[18] Xu, G., Dong, J., Ma, C., Liu, J., & Cliff, U. G. O. (2022). A Certificateless Signcryption Mechanism Based on Blockchain for Edge Computing. IEEE Internet of Things Journal.

[19] Asheralieva, A., &Niyato, D. (2022). Secure and Efficient Coded Multi-Access Edge Computing with Generalized Graph Neural Networks. IEEE Transactions on Mobile Computing.

[20] Kubiak, K., Dec, G., &Stadnicka, D. (2022). Possible Applications of Edge Computing in the Manufacturing Industry—Systematic Literature Review. Sensors, 22(7), 2445.

[21] Lin, Y., Mao, Y., Zhang, Y., & Zhong, S. (2022). Secure Deduplication Schemes for Content Delivery in Mobile Edge Computing. Computers & Security, 102602.

[22] Ming, Y., Wang, C., Liu, H., Zhao, Y., Feng, J., Zhang, N., & Shi, W. (2022). Blockchain-Enabled Efficient Dynamic Cross-Domain Deduplication in Edge Computing. IEEE Internet of Things Journal.

[23] Du, J., Han, G., & Lin, C. (2022). An Edge-Computing-Enabled Trust Mechanism for Underwater Acoustic Sensor Networks. IEEE Communications Standards Magazine, 6(1), 44-51.

[24] Saravanakumar, S., & Saravanan, T. An effective convolutional neural network-based stacked long short-term memory approach for automated Alzheimer's disease prediction. Journal of Intelligent & Fuzzy Systems, (Preprint), 1-16.

[25] Li, T., He, X., Jiang, S., & Liu, J. (2022). A survey of privacy-preserving offloading methods in mobile-edge computing. Journal of Network and Computer Applications, 103395.

[26] Wang, R., Lai, J., Zhang, Z., Li, X., Vijayakumar, P., & Karuppiah, M. (2022). Privacy-Preserving Federated Learning for Internet of Medical Things under Edge Computing. IEEE Journal of Biomedical and Health Informatics.

[27] Gupta, S. (2022). Non-functional requirements elicitation for edge computing. Internet of Things, 18, 100503.

[28] Tulkinbekov, K., & Kim, D. H. (2022). Blockchain-enabled Approach for Big Data Processing in Edge Computing. IEEE Internet of Things Journal.

[29] Kumar, A., Upadhyay, A., Mishra, N., Nath, S., Yadav, K. R., & Sharma, G. (2022). Privacy and Security Concerns in Edge Computing-Based Smart Cities. In Robotics and AI for Cybersecurity and Critical Infrastructure in Smart Cities (pp. 89-110). Springer, Cham.

[30] Dong, J., Zheng, F., Lin, J., Liu, Z., Xiao, F., & Fan, G. (2022). EC-ECC: Accelerating elliptic curve cryptography for edge computing on embedded GPU TX2. ACM Transactions on Embedded Computing Systems (TECS), 21(2), 1-25.

[31] Yu, X., Zhao, W., & Tang, D. (2022). Efficient and provably secure multi-receiver signcryption scheme using implicit certificate in edge computing. Journal of Systems Architecture, 126, 102457.

[32] Souza, P. S., Ferreto, T. C., Rossi, F. D., &Calheiros, R. N. (2022). Location-Aware Maintenance Strategies for Edge Computing Infrastructures. IEEE Communications Letters, 26(4), 848-852.

[33] Wan, Y., Qu, Y., Gao, L., & Xiang, Y. (2022). Privacy-preserving blockchain-enabled federated learning for b5g-driven edge computing. Computer Networks, 204, 108671.

[34] Sun, Q., Wu, H., & Zhao, B. (2022). Artificial intelligence technology in internet financial edge computing and analysis of security risk. International Journal of Ad Hoc and Ubiquitous Computing, 39(4), 201-210.

[35] Bandi, A. (2022, March). A Review Towards AI Empowered 6G Communication Requirements, Applications, and Technologies in Mobile Edge Computing. In 2022 6th International Conference on Computing Methodologies and Communication (ICCMC) (pp. 12-17). IEEE.

[36] Zhang, W., & Zhong, S. (2022). Data Legal Supervision of Online Car-Hailing Platform Based on Big Data Technology and Edge Computing. Wireless Communications and Mobile Computing, 2022.

[37] Choi, J. H. (2022). Key-Agreement Protocol between IoT and Edge Devices for Edge Computing Environments. Journal of Convergence for Information Technology, 12(2), 23-29.

[38] Yi, Z., Wei, L., Yang, H., Wang, X. A., Yuan, W., & Li, R. (2022). An Improved Secure Public Cloud Auditing Scheme in Edge Computing. Security and Communication Networks, 2022.

[39] Dhanare, R., Nagwanshi, K. K., & Varma, S. (2022). A Study to Enhance the Route Optimization Algorithm for the Internet of Vehicle. Wireless Communications and Mobile Computing, 2022.

[40] Joloudari, J. H., Alizadehsani, R., Nodehi, I., Mojrian, S., Fazl, F., Shirkharkolaie, S. K., & Acharya, U. R. (2022). Resource allocation optimization using artificial intelligence methods in various computing paradigms: A Review. arXiv preprint arXiv:2203.12315.