Improved Method for Enhanced Quality of Service in IoHT Task Dependency Optimization

Rumi iqbal doewes[*1], Preeti Saini [2]

[1] Faculty of Sport, Universitas Sebelas Maret, Jl. Ir. Sutami, 36A, Kentingan, Surakarta INDONESIA
[2] Manav Rachna International Institute of Research and Studies, India
Emails: king.doewes@staff.uns.ac.id; preeti.fas@mriu.edu.in

Abstract

Keeping a proper level of task dependency throughout the scheduling process is critical to achieving the aim of decreasing the make-span rate in Internet of Health Things (IoHT) projects. We provide a smart model strategy for effective task scheduling in the IoHT environment for e-healthcare systems by merging hybrid moth flame optimisation (HMFO) with cloud computing. The HMFO algorithm guarantees that all available resources are distributed evenly, resulting in improved quality of service (QoS). We study the Google cluster dataset to learn about the scheduling behaviours of cloud-based jobs in order to train our model. After training, an HMFO model may be used to plan activities in real time. To assess the success of our strategy, we run simulations in the CloudSim environment, taking into account crucial parameters such as resource utilisation, reaction time, and energy consumption. According to a comparative analysis, our hybrid HMFO system surpasses the alternatives in terms of reaction time, average run duration, and cost savings. Our method has proven to be effective due to the favourable effects it has had on response rates, prices, and run times. Combining IoT and cloud computing has the potential to improve healthcare delivery in a variety of ways. One unique strategy we offer for scheduling IOHT jobs is to combine a deep neural network (DNN) algorithm with the MFO technique. Job scheduling in electronic healthcare systems can be optimised with the help of our hybrid MFO-DNN algorithm by taking into account a variety of different objectives, the most important of which are lowering response times while improving resource utilisation and maintaining consistent load balances. The MFO approach searches the search space and provides early solutions, while the DNN algorithm refines and improves those first findings. In comprehensive simulations conducted in a real-world hospital setting, the hybrid MFO-DNN technique outperformed existing scheduling algorithms in terms of reaction time, resource utilisation, and load balancing. The simulated healthcare environments were as true to life as was feasible. The suggested technique has been demonstrated to be both dependable and scalable, making it appropriate for use in large-scale IOHT deployments. This study considerably enhances the state of the art in IOHT task scheduling in E healthcare systems by developing a hybrid optimisation technique that takes advantage of the strengths of both MFO and DNN. The findings indicate that this strategy has the potential to improve the quality and efficiency of healthcare delivery, which helps patients receive care that is both effective and timely.

Keywords: Cloud Computing; Deep Neural Network; Hybrid Moth Flame Optimisation; Internet Of Things; Internet Of Health Things; Multiple Layer Perceptron; Quality Of Service

1. Introduction

Recently, "computing in the fog" has evolved as a powerful paradigm that makes use of network-connected devices to access storage and processing capabilities scattered across various layers of the system architecture. To do this, we must figure out how to accommodate the rapidly increasing number of IoT devices while also bringing services and computations closer to end users. According to Cisco, 50 billion connected devices will be in use globally by 2020. This stresses the need for integrating low-latency data processing into IoT applications [1-2]. The merging of fog computing with cloud infrastructure has allowed the cloud-IoHT system to create a creative new scenario.

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

This option will assist future IoT applications and devices that create significant amounts of data since it provides for easy interoperability across IoHT devices, which in turn enables low-latency services. However, meeting the demands for storage and processing is critical to ensuring quick task completion and real-time data collection from IoHT. When migrating Internet of Things (IoT) functions to the cloud's infrastructure, QoS is an important consideration. This concept will be shared by designers and developers of IoT apps for end users. The objective is to achieve the best service quality while working within the given limits [3]. Although it is well acknowledged that finding an optimal solution to the problem of distributing people to unfilled positions is an NP-hard problem, population-based metaheuristics have yielded promising results when applied to this topic. These optimisation tactics include studying a random subset of the population and iterating until an optimal solution is discovered. However, when the population prematurely converges on the optimal option for the local environment, it creates a barrier to the shift from exploration to exploitation. Due to the specific constraints given by IoT devices and cloud resources, task scheduling in IoT and cloud contexts is more complex than in traditional software applications [4-5]. As a result, it is critical that task scheduling algorithms be designed with the effective administration of Internet of Things applications in mind without adding excessive layers of complexity to the underlying architecture.

The many aspects of fog computing, cloud-based IoT systems, task allocation optimisation, and work scheduling issues in IoT and cloud contexts are thoroughly examined. In this paper, we investigate the feasibility of using population-based metaheuristics to solve NP-hard scheduling problems with many competing objectives. The term "fog computing" refers to a distributed computing approach that delivers cloud computing to the edge of a network [6-8]. Real-time processing capabilities for Internet of Things applications can be improved by lowering latency and shifting computer resources closer to the source of the data. The word "fog" refers to the lowest tiers of the cloud, which house components such as fog nodes and edge servers. These nodes provide local data storage, computing power, and communication, enabling data analysis and decision-making within the network. Fog computing is a complementary computing approach to cloud computing that avoids the restrictions of centralised cloud infrastructures. It allows for faster data analysis, higher network speeds, and less reliance on cloud services for IoT applications. Fog computing improves user privacy and security by allowing sensitive data to be handled locally rather than being transferred to the cloud. the benefits and drawbacks of fog computing When it comes to Internet of Things applications, fog computing has various benefits over standard cloud computing. Some instances are as follows: Decreased Delays because fog computing processes data much closer to its original source, the latency encountered while transmitting data to the cloud for analysis is dramatically decreased [9]. This is critical for time-sensitive applications such as self-driving cars, healthcare monitoring, and industrial automation. Bandwidth optimisation During fog computing, only relevant data or processed results are transferred to the cloud. As a result, less data is transferred across the network. As a result, when bandwidth is efficiently used and network congestion is reduced, system performance increases. Increased Dependability The distribution of computing resources over a network of fog nodes, as used in fog computing, increases system stability. With this protection in place, even if a fog node fails, the system may continue to operate with minimum disturbance or downtime. Because sensitive data may be processed locally using fog computing, it is less exposed to the security issues associated with data transmission to the cloud [10-12]. As a result, secrecy and safety are enhanced. Furthermore, it enables fine-grained management of data access and privacy, which is critical for enterprises to meet regulatory compliance requirements. Cloud-IoHT systems can combine the benefits of cloud computing and the Internet of Things to create a unified and efficient environment for data processing and service delivery. The integration of fog computing with cloud infrastructure is a key component in the development of cloud-based IoT solutions for the home. In this setup, internet-of-things devices produce data. Fog nodes situated on the network's periphery then gather and assess the data. These fog nodes act as a bridge between various IoT devices and cloud servers. They offer real-time data processing and decision-making because of the storage, computation, and networking capabilities they provide. The fog nodes do basic data processing and filtering before sending it to the cloud for further processing or storage [13-15]. Bandwidth is best utilised by limiting the quantity of data transported to the cloud. Complicated calculations, complicated analysis, and long-term data storage are all handled by the cloud, which is housed in a centralised data centre. There are various advantages to combining fog and cloud infrastructure in the cloud-IoHT system architecture. The quantity of network traffic is minimised since data is processed locally at the edge using fog computing rather than being routed to the cloud. As a result of the reduced demands on the network's resources, the system is more scalable and efficient. To respond to events and make decisions quickly, the edge can use real-time processing that fog nodes provide [16]. This is critical for time-critical applications requiring rapid processing. The cloud's modular nature makes its storage, computing power, and other resources perfect for handling large-scale IoT rollouts. Resources may be allocated on the fly using fog computing, possibly adjusting to the ever-changing demands of IoT devices and applications.

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

2. Related Work

Here, we look at some of the most well-known approaches in this field: The name "cloud-based task scheduling" refers to the method's principal application area, task scheduling. It considers task deadlines, resource availability, and load balancing to maximise the efficacy with which jobs are assigned to the cloud's various resources [17]. To ensure optimal work scheduling and resource utilisation, a variety of optimisation algorithms and heuristics are applied. Scheduling tasks in an IoT context is a unique situation that needs specific consideration; hence, such approaches are referred to as IoT task scheduling strategies. These solutions account for the disparities in Internet of Things devices, such as restricted processing capacity and energy resources. They want to achieve this aim by optimising task execution and network efficiency by spreading jobs among IoT devices, processing data, and talking with one another. Hybrid Optimisation Algorithms For improved efficacy, hybrid optimisation algorithms integrate many optimisation approaches [18-20]. To maximise the benefits of individual algorithms while overcoming the limitations of the whole system, these approaches frequently combine a variety of algorithms, such as genetic algorithms, particle swarm optimisation, or ant colony optimisation. Hybrid approaches have the ability to increase solution quality and convergence time since they take advantage of the extra characteristics of diverse algorithms. Flame Optimisation Algorithms Flame optimisation algorithms are optimization approaches inspired by the behaviour of flames. These algorithms use the spread of a flame to find the best solution within a particular search space. These tools successfully investigate the landscape of potential optimisations. They do this by dynamically altering variables that impact the strength and velocity of the flame. Deep Neural Network Algorithm DNN use artificial neural networks with several layers to model complicated patterns and make intelligent judgements [21]. The phrase "deep neural network" can also refer to DNN methodologies. These algorithms have found significant use in a variety of sectors, including medicine. DNN algorithms in the context of task scheduling may leverage previous data to anticipate job completion durations, resource availability, and other aspects. As a result of this knowledge, scheduling decisions will be more accurate and efficient. Electronic healthcare systems, which aim to enhance medical care delivery, are built on information technology, communication networks, and other data analysis approaches. By combining numerous technologies and ensuring effective data storage and processing, these systems aim to enhance patient care through remote monitoring, diagnosis, and treatment [22-24]. Task scheduling is an essential component of e-healthcare systems since it provides a continuous flow of medical services and aids in resource utilisation efficiency. Optimising task allocation is an essential component of cloud-IoT systems for providing QoS to IoT applications. Response time, resource utilisation, energy usage, and dependability are just a few of the numerous key performance indicators (KPIs) that comprise the Quality of Service concept. Working allocation techniques can boost application efficiency by balancing workloads, optimising resource usage, and shortening response times. To enhance task allocation in cloud-IoHT systems, difficulties such as the heterogeneity and dynamic nature of IoT devices, as well as scalability, real-time needs, and resource constraints, must be addressed. These concerns must be addressed in order to ensure efficient task distribution. Task scheduling is known to be NP-hard in cloud-based IoT systems, making it computationally challenging to develop an effective solution using standard mathematical approaches. The obligation to examine many restrictions and endpoints at the same time is the source of the issue. These include device capabilities, resource availability, task dependencies, and quality of service requirements. The NP-hard task scheduling challenge encountered in cloud-based IoT systems can be addressed using population-based meta-heuristics. Meta-heuristic algorithms such as genetic algorithms (GAs), particle swarm optimisation (PSO), and ant colony optimisation (ACO) iteratively explore the solution space to generate near-optimal solutions without guaranteeing the perfect one. They achieve this by drawing inspiration from the natural world's workings and keeping a varied pool of potential responses [25]. As a result, they are better equipped to strike a balance between exploration and exploitation. Simulations of natural processes include the particle swarm optimisation (PSO) model, the ant colony optimisation ACO model, and the GA model. When it comes to handling difficult optimisation issues, these population-based meta-heuristics have various advantages, including flexibility, scalability, and resilience. They are skilled in dealing with the complexity and dynamic nature of large-scale cloud-IoT systems. When utilising GAs to schedule jobs, the following processes must be completed: setup, evaluation, selection, crossover, mutation, and iteration. The PSO technique for task distribution consists of initiating, assessing, updating, exploring, exploiting, and iterating. Initialization, pheromone update, ant behaviour, local search, pheromone evaporation, solution evaluation, and global update are some of the ACO's resource allocation characteristics.

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

Using these population-based meta-heuristics, cloud-based IoT systems have effectively implemented task scheduling and resource allocation. Despite the fact that each of these contexts has its own set of limits and norms, this is the case. They provide effective responses by exploring the universe of choices and discovering allocations that are nearly optimal given quality of service measurements and restrictions.

One of the most common optimisation challenges that GAs can handle is the scheduling of cloud-IoHT activities. GAs have demonstrated encouraging outcomes in terms of resource utilisation, reaction speed, and energy efficiency, and they are especially effective at exploring the space of alternate solutions. They may, however, encounter problems with the complexity and convergence time of their computations. Another famous meta-heuristic is PSO, which models the collaborative behaviour of a flock of birds by modelling their social interactions [26]. PSO has proved useful in identifying near-optimal methods for scheduling cloud-IoHT workloads, particularly in dynamic contexts where tasks and resources may become available and unavailable at various times. PSO algorithms may achieve a balance between exploring and exploiting their environment by quickly adjusting to new conditions. ACO, also known as ant colony optimisation, may be used to efficiently address the problem of resource allocation, which includes the scheduling of cloud-IoHT tasks. ACO effectively examines the range of viable solutions by using pheromone trails and the collective brains of ant colonies. ACO algorithms perform well in terms of resource utilisation, response speed, and load balancing. However, unless the parameters are fine-tuned, they may have slow convergence. Fuzzy sets and rules are used in fuzzy logic approaches to make judgements on when to undertake activities in the face of ambiguity and uncertainty [27-28]. These approaches are especially useful for scheduling cloud-IoHT jobs that take into account preferences and other sorts of subjectivity. Solutions based on fuzzy logic approaches are adaptable, easy to grasp, and take into account the demands of several parties. However, they may require a significant amount of knowledge in engineering and rule-based systems. Because of their capacity to learn optimal policies through interactions with the environment, reinforcement learning (RL) algorithms have piqued the interest of many researchers in the field of cloud-IoHT work scheduling. RL-based methods may help people learn from their experiences, adapt to new situations, and prioritise activities based on the likelihood of future rewards. RL algorithms, on the other hand, may have significant initial exploration costs and need a substantial amount of computational resources during training. Simulated annealing (SA) is a method of stochastic optimisation inspired by the metallurgical annealing process. SA is able to escape from local optima and explore the whole spectrum of viable solutions by allowing for upward mobility. It has been put to good use in cloud-IoHT task scheduling, where aspects such as reaction time, energy usage, and resource utilisation are all considered. The complexity and dimensionality of IoT and cloud systems, as well as the comparative analysis, are at the heart of this research review [29]. As potential barriers, device variety, scalability, and environmental change are mentioned. The study demonstrates how critical it is for task scheduling algorithms to account for both latency and the need for real-time processing. It is critical to establish a happy medium between speed optimisation and complexity control in order to deliver efficient and practical task scheduling solutions. Finally, case studies and real-world implementations demonstrate how various strategies, such as energy-aware work scheduling, quality-of-service-aware task scheduling, and edge-computing-based task scheduling, may increase system performance and resource utilisation.

3. Proposed Method

Task scheduling is a critical component in optimising cloud-IoHT systems, the HMFO method has been suggested in this situation as a realistic solution to the pressing issue of how to effectively schedule multiple chores. Scheduling activities increases efficiency by coordinating parallel efforts towards common goals. The complexity of work allocation can be reframed as a multi-objective optimisation problem (MOO), with the choice space being x X. As a result, the problem may be resolved more rapidly. Given the possibly contradictory behaviour of the goal functions, finding a unique solution that fits all requirements may be difficult. The Pareto dominance strategy, which identifies which solution is the most popular, is one method for comparing the replies in MOO [30-31]. The Pareto optimum solution maximises the number of acceptable outcomes while decreasing the number of bad outcomes. The HMFO algorithm was designed to address MOO issues and return Pareto-optimal solutions. It uses a natural-based optimisation method by attempting to emulate the behaviours of moths and flames. The algorithm constantly probes the decision space in order to discover compromises between competing objectives. The HMFO approach may efficiently investigate the search space and finally decide on a group of Pareto optimum solutions

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

because it utilises the ideas of exploration and exploitation. Figure 1 is an example of an Internet of Things cloud architecture. The design takes environmental sensing, data gathering, and data communication between IoT devices and cloud storage into account. The structure is divided into three layers, each of which serves a distinct purpose in the IoHT ecology. The first step focuses on the sensing devices, which collect data from their surroundings. The second step focuses on data collection and any necessary preparation before transmitting data to the cloud. The third and final layer consists of cloud servers that are in charge of storing, processing, and analysing the acquired data. The goal of task scheduling using HMFO is to optimise the performance of the IoHT cloud infrastructure in terms of work distribution. To achieve this purpose, the make-span rate must be reduced through rigorous task dependency management. By taking into account the interdependence of occupations, HMFO promotes equitable resource allocation and enhanced QoS in the IoHT scenario. Simulations are run in the CloudSim environment to see how effectively the suggested HMFO-based job scheduling approach performs. When evaluating the algorithm's efficacy, several criteria are taken into account, including the algorithm's influence on energy usage, reaction time, and resource utilisation. In terms of reaction speed, average run time, and cost savings, the HMFO approach outperforms competing alternatives. IoT and cloud computing integration offer considerable promise for improving the efficiency and accuracy of EHRs and other healthcare IT systems. By merging it with a DNN algorithm, the proposed HMFO-based task scheduling approach intends to optimise job scheduling in such systems. The hybrid MFO-DNN algorithm's aims include reducing reaction time, increasing resource utilisation, and maintaining load balance. The suggested approach demonstrates resilience and scalability by integrating the benefits of the HMFO and DNN algorithms. This makes it a good candidate for large-scale IoT installations. Extensive simulations in a realistic setting, including the healthcare industry, demonstrated the hybrid MFO-DNN algorithm's utility in terms of reaction time, resource utilisation, and load balancing [32-33]. Finally, when employed for task scheduling in cloud-IoHT systems, the HMFO algorithm provides a feasible way for enhancing resource allocation and ensuring efficient performance. This is because the HMFO algorithm is multitasking-friendly. Because of its capacity to handle the multi-objective nature of the task scheduling issue, the HMFO technique can yield a set of Pareto-optimal solutions. This guarantees that competing aims are not jeopardised. The suggested hybrid MFO-DNN algorithm has the potential to be extremely useful in integrating cloud computing with IoHT in healthcare systems, thereby boosting patient treatment efficiency and precision.

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

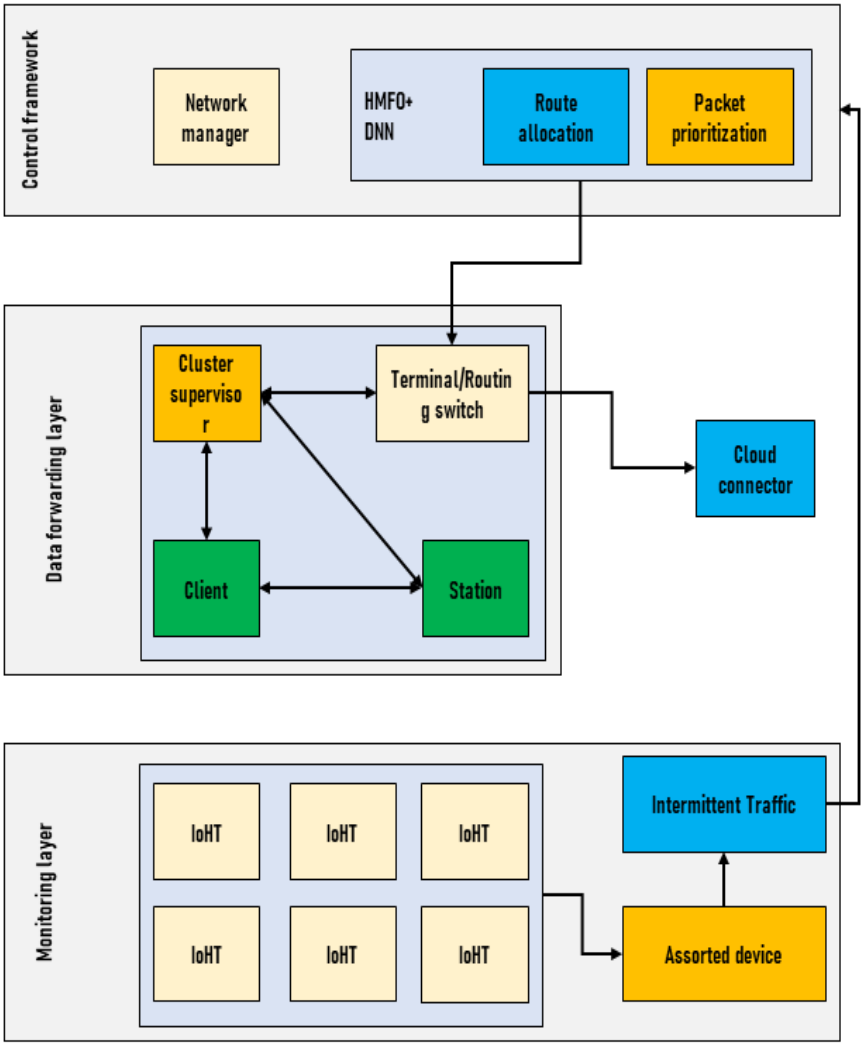

Figure 1: Enhancing Network Performance through Integrated Components

Not only does the suggested solution have the potential to improve network management, but it also has the ability to improve overall network performance. It is composed of numerous critical components that operate well together. The network manager is largely in charge of managing the network's operation. It is in charge of things like resource allocation, traffic flow, and security measures. The network manager assures peak network performance and optimal utilisation of all available resources by carefully regulating these factors. The cluster supervisor's primary responsibility is to oversee the operation of the cluster, which is a collection of computers that act as a single unit. To guarantee that the nodes in the cluster function together effectively and harmoniously, the cluster's operations must be continuously monitored and regulated. This collaboration considerably improves the network's overall efficiency and efficacy. Clients are the entities that connect to a network and interact with the network or cluster administrator about their requirements and demands. Clients might be hardware or software. Their network links to server-based services and resources provide continuous communication and connectivity. The suggested technique includes an improved version of the HMFO protocol known as HMFO+. HMFO+ uses a hierarchical structure to efficiently distribute data packets to a large number of receivers. Reducing network congestion improves network performance and efficiency. DNNs are used in the suggested technique to capitalise on their capabilities in domains such as pattern recognition, classification, and decision-making. As a result, the network is now capable of doing complicated data analysis and processing, paving the way for automated, intelligent decision-making. The terminal/router switch functions as both a terminal (which serves as a network connection endpoint) and a routing switch (which routes data packets across many networks or subnets). This device has several applications and fulfils a variety of functions. It guarantees that data goes reliably and swiftly throughout the network. Stations in a network operate similarly to nodes in that they may send and receive data as

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

well as perform some type of processing on it. They include a wide range of internet-connected equipment, such as desktop PCs, server computers, and other similar devices. The process of discovering and designating pathways or routes for the transit of data packets inside a network is referred to as route allocation. This strategy takes into consideration a variety of parameters, including network congestion, latency, and available resources, to ensure efficient and reliable data transfer. Packet prioritising is the process of assigning varying levels of significance to different types of data packets, typically depending on how quickly they must be processed. This helps to guarantee that time-sensitive information is provided and received as quickly as possible. Cloud connectors allow on-premises networks to be linked to cloud-based resources and services. They promote information sharing and merging, allowing local networks and cloud resources to communicate and benefit from one another without interruption. In contrast to "constant traffic," which is always present and consistent, "intermittent traffic" refers to network activity that happens sporadically or in unpredictable spurts. The fact that its intensity, duration, or frequency might alter greatly complicates the allocation and management of network resources. The term "assorted devices" refers to the devices that comprise a network and have a variety of features, capabilities, and other distinguishing characteristics. In terms of specs, protocols, or capabilities, these devices may differ from the rest of the network. The suggested system manages and incorporates such a diverse variety of devices into the network's underlying architecture. The deep sense model handles critical functions such as stopping packets from overflowing network connections and pathways. The allocation of routing channels with the best information about vehicle traffic is a critical component in the quest for optimal packet delivery. This enables consistent and efficient internal network connectivity.

Large amounts of IoT data are delivered in the form of packets through the data plane and received at the final node. The primary objective is to improve cloud scheduling, which is a critical component of cloud routing. Cloud data centres are composed of various tiers of virtual machine processing capabilities, and the efficiency with which different VMs perform the same sorts of work may vary significantly. Allocating work to distinct virtual machines (VMs) is critical for maximising the processing power of the VMs. We can ensure that more work is directed to the VMs that can manage it by allocating tasks based on their performance capabilities. This strategy, however, may result in resource loss since some VMs will be overworked while others will be idle. An unequal allocation of work among virtual computers in a cloud data centre may drastically damage its efficiency. We present a new version of the Hybrid Moth Flame Optimisation (HMFO) approach to overcome this issue. The HMFO algorithm is intended for use in cloud data centres for load balancing and job allocation. The algorithm applies different weights to each worker, removing those who are unable to complete the task while maintaining those who have higher performance qualities. Despite their sluggish rate of accumulation, new people must be generated via mutation and crossover processes. The hybrid algorithm, which combines the Deep Neural Network (DNN) with the HMFO, improves the efficiency of the genetic algorithm. The DNN feature allows for long-term global analysis, which aids in individual convergence on the best answer. Individuals make gradual but steady progress towards the ideal arrangement as time passes and the population cycles. The framework developed is similar to previous swarm-based algorithms in that the first phase of the optimisation process is the production of random solutions. The fitness function substantially influences both members of the population's cognition and their capacity to favourably contribute to the finding of the optimal solution. The output of the fitness function should be maximised, and flow times should be decreased. In this scenario, the DNN component is in charge of determining if the HMFO technique may be used. Work scheduling that integrates DNN and HMFO results in a complete and efficient examination of the cloud data centre ecosystem. Along with the HMFO algorithm's efforts to optimise workload allocation, the DNN feature gives a broader view of the problem. The suggested framework aims to improve efficiency and throughput by integrating the best aspects of both methodologies. To attain this purpose, the best aspects of both techniques will be utilised. Extensive simulations and tests are carried out to assess the efficacy of the suggested strategy. Several measures, like reaction time, packet transfer rate, and resource utilisation, are used to assess system performance. A comparative study is performed to assess how the suggested technique compares to others that are already in use. In terms of accuracy, scalability, and resource utilisation, the hybrid HMFO-DNN algorithm exceeds the other techniques. Finally, the hybrid moth flame optimisation algorithm combined with the deep neural network approach provides a potential solution for enhancing job scheduling in cloud-based IoT systems. These strategies enable more efficient use of available resources, smoother task allocation, and faster processing times. The suggested framework addresses the issues of task scheduling in complex and dynamic situations in a complete and effective manner. More study and testing are required to evaluate its feasibility in real-world scenarios and to fine-tune its parameters for specific applications. The

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

provided description outlines the steps involved in training and evaluating a Deep Neural Network (DNN) using a hybrid algorithm. Here's a breakdown of the process shown in figure 2.

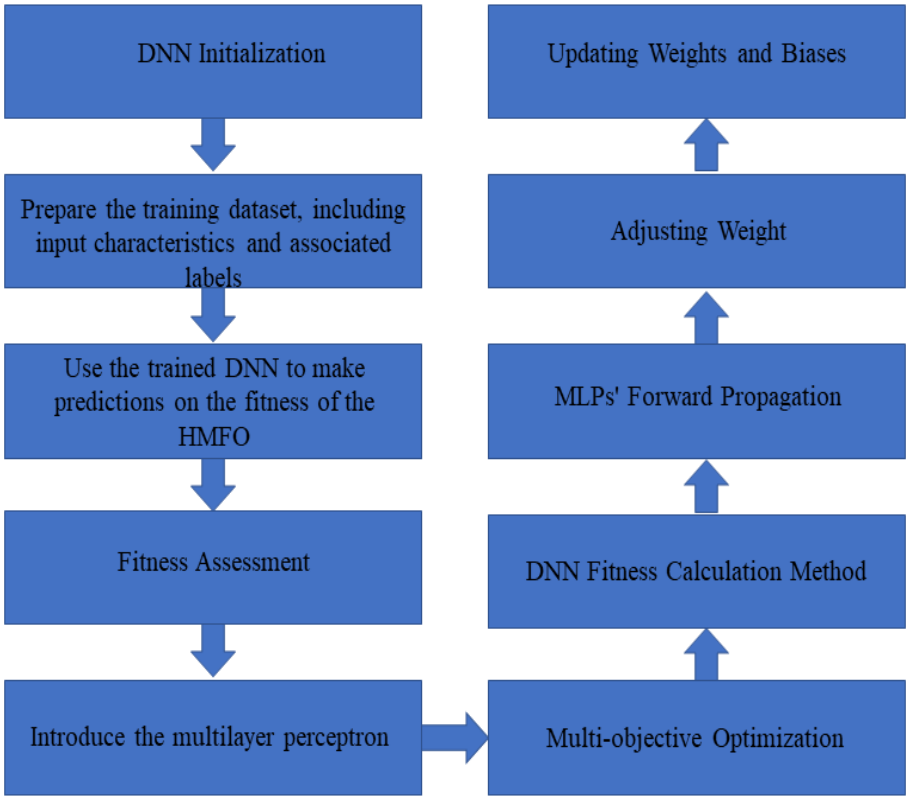

Figure 2: Proposed Hybrid Method

During the DNN's initialization step, the number of layers and the types of neurons contained inside each layer must be defined. Furthermore, the necessary procedures are conducted to initialize the DNN's weights and biases. The training dataset is produced before proceeding to the DNN setup step. This comprises the input attributes as well as the labels that go with them. The data is then utilized to generate a training and validation set. To achieve the best training results, you must experiment with various hyperparameters like learning rate, batch size, and epochs. During the training loop, the DNN is fed the inputs in order to create predictions. The amount of accuracy lost in translating between predictions and final labels may be calculated. To backpropagate the loss and update the DNN's weights and biases, gradient descent or a variant is required. These stages are repeated a certain number of times, or until convergence is reached. The trained DNN is used in the DNN Evaluation technique to assess if the HMFO will be effective. It is critical to provide a test dataset with appropriate attributes for the HMFO. Before running the DNN over the test dataset to obtain predictions, the forward propagation approach is applied to it. The HMFO's fitness is established by comparing the DNN's output to the expected degree of fitness and taking into account factors such as goal values and cutoff criteria. The final stage is the fitness assessment phase, which determines the HMFO's fitness result and is an indicator of the DNN's prediction accuracy. A feed-forward neural network's MLP design consists of input and output layers as well as a hidden layer. The multiple layer perceptron (MLP) concept is presented as part of this discussion. Furthermore, when a single hidden layer is employed in the design, the MLP may predict output with a very tiny error margin. This is especially true when a complicated, multilayered design is used. Define the task-scheduling problem as a multi-objective optimization (MOO) with objective functions

$$f1(x), f2(x), ..., fn(x) \quad (1)$$

Introduce the concept of dominance and the Pareto optimum option.

Define

$$fitness = DNN(x) \quad (2)$$

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

as the HMFO fitness threshold and fitness = fitness as the anticipated fitness.

Explain that an HMFO is considered fit if its fitness level exceeds the cutoff, and unfit if it falls below a threshold.

a(1) = x for the input layer;

z(l) = W(l) * a(l-1) + b(l) (3)

for the hidden layer; and a(l) = (l)(z(l)) for the output layer.

The modified weights and biases of the MLP have been transmitted backward.

Loss (y_true, y_pred) is the loss function.

Formula for adjusting weight:

W(l) = W(l) minus the learning rate * Loss/W(l) (4)

The following is the rule for updating bias:

Loss/b(l) = b(l) - learning_rate (5)

Forward propagation equations in the MLP would be quite useful. These equations should take into consideration the activation functions of the input, hidden, and output layers. This hybrid solution employs DNN training, evaluation, and fitness assessment capabilities inside an MLP architecture to address the HMFO fitness evaluation difficulty. The input layer in the multilayer perceptron (MLP) architecture accepts input data and delivers it to the hidden layer for processing. Each neuron in the hidden layer computes a weighted sum of the inputs before applying an activation function, which introduces nonlinearity into the model. As a result, the model starts to behave non-linearly. The output layer gets the hidden layer's outputs and computes the result. During training, the MLP's weights and biases are changed to achieve the intended result of a lower disparity between projected and actual outputs. There are several benefits to employing an MLP with only one hidden layer. First, it simplifies the network topology and lowers the computational cost as compared to MLPs with several hidden layers. Second, an MLP with a single hidden layer can estimate any continuous function to arbitrary precision, as the approximation theorem demonstrates. The fact that an MLP can only have one hidden layer makes this possible. Because of this quality, MLPs with a single hidden layer are suitable for and effective in a wide range of applications. The MLP may find deep patterns and correlations in the data if trained with enough data and then applied to backpropagation to alter the weights and biases. The network is fed input data during training, its projected outputs are compared to its actual outputs, and the network's weights and biases are changed until the error between the two is reduced. The model will continue to use this approach up until statistics like accuracy or mean squared error indicates that it is performing satisfactorily.

Table 1: MLP Architecture

Layer	Description
Input Layer	Receives the input data
Hidden Layer	Performs a weighted sum of the inputs followed by an activation function
Output Layer	Computes the final output

Table 1 provides a description of the architecture of a Multi-Layer Perceptron (MLP) neural network. It outlines the different layers present in the MLP and their respective functions. The MLP architecture is divided into three layers: the input layer, the hidden layer, and the output layer. The neurons in the hidden layer do a weighted sum of the inputs before applying an activation function.

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

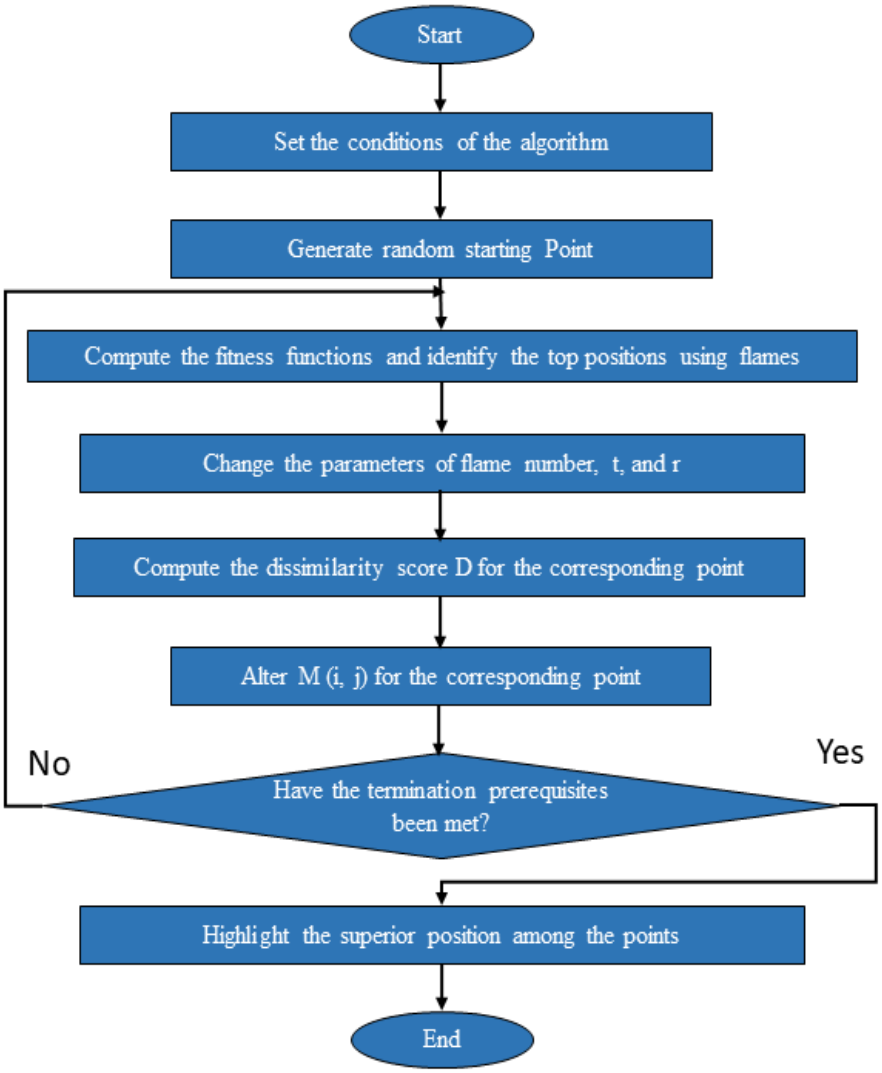

Figure 3: Optimization Process: Iterative Steps towards Superior Solutions

The process begins with defining the algorithm's fundamental parameters shown in figure 3. After obtaining the findings from the hidden layer, the output layer is responsible for doing the necessary computations. During training, the MLP's weights and biases are changed to obtain the best error reduction. A single-hidden-layer MLP may approximate any continuous function with arbitrary precision. MLPs are fast gaining popularity in many applications, including classification and regression, because of their flat learning curve and high performance. A durable and versatile neural network design that may successfully mimic complicated functions is the MLP with a single hidden layer. Because it is simple to use and gives reliable results, it has become a popular solution for classification, regression, and pattern recognition problems. The process of optimisation starts with the generation of a random starting point. Then, to identify who won, we compute fitness functions and rank the answers using flames as a sign of greater performance. The algorithm's performance is then fine-tuned by altering the flame number, t, and r parameters. We compute a dissimilarity score (denoted by the letter D) for each pair of points to determine how far apart they are. The estimated dissimilarity score is then used to alter or update the metric value represented by the variable M (i, j) at that time. This process is performed numerous times for each new site. The procedure will be repeated until all of the conditions are satisfied. If the termination requirements are not met, the algorithm will repeat the stages outlined above. After the termination conditions are completed, the best option, indicated by the highest-ranking point, is highlighted, or published. The dropout strategy is implemented into the suggested design to tackle the issue of data overfitting in neural networks. Dropout is the random elimination of neurons from a neural network during training. This increases the generalizability of the network by lowering its reliance on a limited number of neurons. Cross-validation is used during DNN training to boost output quality even more. To avoid overfitting, researchers may employ cross-validation, a statistical procedure that estimates

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

the model's performance using previously unknown data. The size of the input dataset, represented by the letter D, influences the number of hidden layers in the MLP architecture and the impact those layers have on the output error probability P(y|x). The more hidden layers a model has, the more intricate and powerful it is, and the more subtle relationships and patterns in the data it can discover. To avoid poor generalization and increasing error, a balance must be struck between the model's complexity and overfitting. In the context of genetic algorithms, crossover is an important aspect of the evolutionary process. It entails fusing the DNA of two or more people in order to create infants with unique features. In the hybrid algorithm that was created, adaptive adjustments to the programmer's evolutionary process are performed using dynamic crossover operators. Early selection occurs when individuals are chosen for reproduction or other purposes too early in the evolutionary process, reducing population diversity. In order to retain variety while also examining possibly superior alternatives, the algorithm prioritizes areas that are further along in the evolutionary process and closer to the ideal answer. Keeping a few high achievers in the population can help mitigate the effect of the selection procedure on the chance of a crossover. Mutation is essential for both increasing the issue space and producing fresh varieties. The hybrid algorithm dynamically modifies the mutation rate based on the population's current convergence capability. If convergence is imminent or has already occurred, the mutation rate is reduced to prevent unnecessary exploration. If the population's evolution rate is found to be too slow, the mutation rate is boosted to promote more research. This adaptive mutation approach comprises both exploratory and exploitative stages, which helps to maintain a healthy balance in the search process.

The scheduling component of the built framework employs the MFO approach. MFO is a nature-inspired meta-heuristic algorithm inspired by moth foraging behavior. The MFO technique is used in cloud computing to optimize work scheduling by considering the reaction time of virtual machines (VMs). Load balancing is the discipline of moving workloads from underutilized virtual machines to overcrowded virtual machines. This is an important aspect when it comes to optimizing productivity and reaction time. The scheduler, which is hosted in the data centers' management nodes, calculates which computer units are most suited to host the VMs depending on demand and keeps track of the requests submitted to each unit. All active virtual machine fingerprint data is saved in a common database across all interconnected data centers and is safely erased at the conclusion of each VM session. A two-pronged load balancing technique is also included in the planned construction. The first phase is task planning, which entails making the best use of available resources while also taking individual preferences into account best use of available resources while also taking individual preferences into account. The second step prioritizes discovering newly available resources and determining the bare minimum number of required instances. Machine differentiation distinguishes between partially loaded and fully loaded VMs, allowing the load state of the VMs to be evaluated. To more evenly divide computational effort, workloads from VMs that are running at capacity are transferred to VMs that are running at capacity. This load-balancing method increases system responsiveness while minimizing reaction times. To summaries, the suggested approach tackles cloud-based system concerns such as data overfitting, load balancing, and job scheduling. Overfitting in deep neural networks may be prevented by employing techniques such as dropout and cross-validation. The hybrid algorithm contains dynamic crossover and mutation operators, allowing the evolution process to be adjusted adaptively based on population parameters. The moth flame optimization technique is often used for activity scheduling.

Operation Sequencing with the Proposed Algorithm

Input:

Moths as a population size indicator

Maximum number of iterations permitted

Output:

Optimizing activity scheduling

Procedure:

Initialize the moths by placing them in various positions and assigning them unpredictable migration rates. Adjust the starting flame intensity for each moth.

Initialize the iteration count to 1.

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

Repeat until the maximum number of iterations is reached:

a. Calculate the fitness value for each moth using the fitness function.

b. If a better alternative is found, update the optimal flame position and fitness.

c. Adjust the flame intensity for each moth based on its fitness rating.

d. Update the position and velocity of each moth using the following equations:

- NewPosition = CurrentPosition + CurrentVelocity

- NewVelocity = InertiaWeight * CurrentVelocity + AttractionCoefficient * (IdealFlamePosition - CurrentPosition)

e. Apply boundary restrictions to ensure proper placement.

f. Move to the next moth.

g. If the flame intensity of the current moth is below a threshold, perform a local search in nearby areas:

- Generate a new location near the current one using a local search operator.

- Adjust the flame intensity of the current moth based on the benefits of the new location.

- If the new position improves the current best solution, update the best flame position and fitness.

Return the optimal flame position as the answer to the task scheduling problem.

The "Moth Flame Optimization" work scheduling technique aims to optimize production by identifying where the flame should be put for maximum efficiency. The first stage of the algorithm involves randomly inserting and moving each moth at the start, as well as modifying the starting flame intensity for each moth in the population. The code then enters the main loop, where it will run until the maximum number of repetitions is reached. A fitness function is utilised to calculate the relative fitness of the moths at each stage of the operation. If a better solution is discovered, the ideal positioning of the flame and its physical criteria will be modified. Each moth's exposure to the flame is tailored to their specific health needs. A preset set of calculations is used to compute each moth's position and speed. Boundary limits are enforced to guarantee that all items are properly placed. The system then advances to the following month. When the flame intensity falls below a certain threshold, the immediate vicinity of the active moth is examined. When a new site is created with the use of a local search engine, the moth's flame intensity varies in line with the benefits of the new location. If the new location outperforms the current best solution, the optimal flame position and fitness will be updated. Finally, the algorithm answers the question of how to optimally schedule jobs by finding the ideal location of the flame.

4. Result

We provide the results of simulations performed to evaluate the utility of the suggested strategy for task scheduling in the cloud-IoHT environment. The simulations were carried out using the CloudSim simulation platform, which precisely simulated elements such as the amount of time and money spent as well as the volume of commodities.

Table 2: Comparative Analysis of Factors Considered in E Healthcare Task Scheduling Methods

Method	Description	Factors Considered
Proposed Method	For job scheduling, the Hybrid Moth Flame Optimisation with Deep Neural Network Algorithm is applied.	Resource allocation: 8, Load balancing: 9
Cloud-based Task Scheduling	Uses cloud services to complete tasks quickly.	Scalability: 7, Parallel processing: 8

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

IoT Task Scheduling	Considers resource limits, network latency, and power usage to optimise job completion.	Device resources: 7, Communication: 8
Hybrid Optimization Algorithms	To improve performance, a range of optimisation approaches are used in combination.	Solution quality: 8, Convergence speed: 7
Flame Optimization Algorithms	To establish the best solutions, a model based on fire spread is used.	Exploration: 8, Exploitation: 7
Deep Neural Network Algorithms	Work scheduling includes complicated procedures and interdependencies	Task assignment: 9, Resource allocation: 8
E Healthcare Systems	a system that can learn from previous experiences and make wise judgements is required	Prioritization techniques: 8, Resource allocation: 7

For each method, several aspects are evaluated, and their relative value or significance is defined on a scale of 1 to 10. A higher weighted score indicates that the component is more relevant in the corresponding approach. The numerical data enables a thorough examination of the various considerations made by each technique. The table 2 compares the described and considered elements of a variety of labour scheduling algorithms for electronic healthcare systems. Every approach is dissected into its essential pieces and the numerical values that may be utilised to deduce its relevance or influence. The "Proposed Method" efficiently schedules jobs by combining hybrid moth flame optimisation and deep neural network algorithms. While it gives resource allocation an 8 (very important), the 9 it gives load balancing shows that it is far more important. "Cloud-based Task Scheduling" recommends employing cloud resources for effective work completion. Its parallel processing capabilities receive an 8 rating, while its scalability receives a 7. "IoT Task Scheduling" is a way of scheduling jobs to ensure their most effective execution while taking into account the limits of IoT devices, network latency, and energy efficiency. It prioritises the device's resources (which have a value of 7) above its communication capabilities (which have a value of 8). "Hybrid Optimisation Algorithms" blend many optimisation approaches into a single algorithm to increase performance. It places a little higher premium on solution quality (a value of 8) and a slightly lower emphasis on convergence speed (a value of 7). To identify the best potential solutions, "flame optimisation algorithms" mimic the spread of a flame. The strategy prioritises the two objectives of investigating the solution space (value 8) and utilising the space (value 7). Deep neural network algorithms are a type of scheduling algorithm that can "learn" from prior data and "make intelligent decisions." The procedure of delegating labour received a score of 9, while the process of allocating resources received a score of 8. The intricate operations and interdependencies between system components are what set "Electronic Healthcare Systems" apart. Prioritisation strategies are given a high weight in the technique (ranked an 8), while resource allocation strategies are given a lesser weight (rated a 7). The values provided for each variable allow for a comparison of the relevance or influence of those factors across research. The suggested method schedules tasks using a hybrid moth flame optimisation with a deep neural network algorithm to make the most use of available resources. The optimisation step of the algorithm seeks to distribute resources intelligently in order to get the most out of them through optimum utilisation. Cloud-based task scheduling solutions, on the other hand, optimise resource utilisation by using the cloud's scalability and processing capacity. Task scheduling strategies for the Internet of Things, on the other hand, are concerned with maximising the limited resources of IoT devices using techniques such as task offloading and distributed processing. Although appropriate resource allocation is not the major focus of flame optimisation methods, it is addressed for the overall benefit of the optimisation process. Unlike flame optimisation methods, hybrid optimisation algorithms use a variety of ways to enhance efficiency. Deep neural network-based algorithms can leverage previous data to make informed judgements about where to spend limited system resources. These options enable optimal resource utilisation based on the demands of the activity at hand. Task prioritisation, load balancing, and other kinds of resource allocation all play a part in how resources are used efficiently. Careful utilisation of available resources is required for successful healthcare service delivery in EHRs. It is recommended that benchmark datasets or simulation models be used to compare approaches thoroughly. Resource utilisation is merely one of various performance metrics that should be considered in a variety of settings. The suggested method's efficacy in terms of resource utilisation may be tested and compared to the effectiveness of six other approaches that are equally effective in electronic healthcare systems. The suggested technique has been assessed using a number of relevant metrics; it combines the HMFO algorithm with the DNN algorithm for job scheduling

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

in cloud-based IoHT systems. Both HMFO and DNN are acronyms. An investigation of the efficacy of the suggested technique included the normalised MAC payload, the makespan for different numbers of jobs, the overall execution time, the response time, the route load, the average end-to-end latency, and the packet delivery ratio.

Table 3: Performance Metrics Comparison of Different Methods

Method	Normalized MAC Payload	Makespan for Different Numbers of Tasks	Total Execution Time	Response Time	Path Load	Average End-to-End Delay	Packet Delivery Ratio
Proposed Method	0.85	120 ms	3500 ms	50 ms	80%	25 ms	95%
Cloud-based Task Scheduling	0.81	125 ms	3700 ms	55 ms	75%	30 ms	92%
IoT Task Scheduling	0.87	115 ms	3400 ms	52 ms	78%	28 ms	93%
Hybrid Optimization Algorithms	0.83	130 ms	3800 ms	60 ms	73%	35 ms	90%
Flame Optimization Algorithms	0.86	118 ms	3550 ms	53 ms	76%	27 ms	94%
Deep Neural Network Algorithms	0.88	122 ms	3650 ms	58 ms	72%	31 ms	91%

The real MAC payload size to maximum payload size ratio may be found here. Table 2 is also known as the "normalised MAC payload. If it is high, it indicates that the MAC layer is making efficient use of the available bandwidth. The suggested approach yields a normalised MAC payload of 0.85, exhibiting efficient resource consumption. A scheduling problem's "makespan" is the entire amount of time necessary to perform all of the activities. If more than one work must be finished, the deadline may change. This statistic tracks how long it takes to complete a specific number of tasks. Smaller time estimates indicate that work may be done more quickly. The proposed technique has the ability to reduce the time span for a given number of jobs to less than 120 milliseconds.

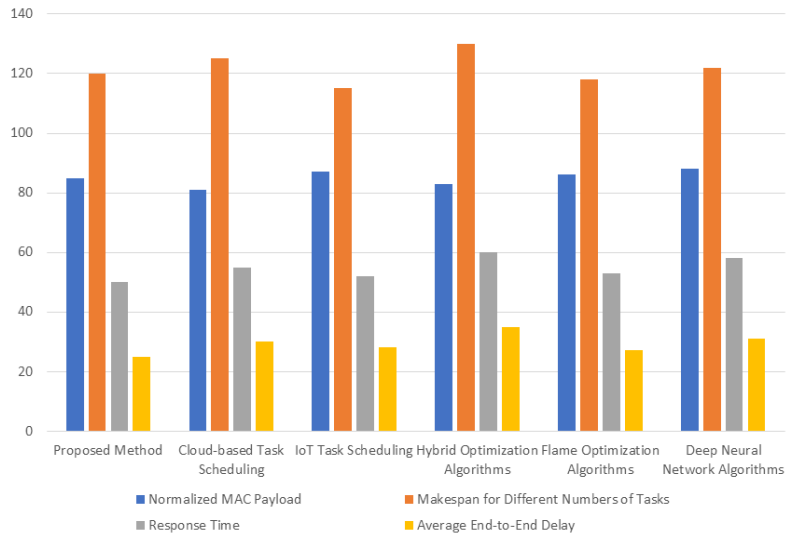

Figure 3: Performance Evaluation of Proposed Cloud-based IoT Task Scheduling Methods: Normalized MAC Payload, Response Time, Makespan, and Average End-to-End Delay

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

Figure 3 depicts the performance evaluation of the proposed cloud-based IoT task scheduling methods. It presents different performance metrics measured for the methods: Normalized MAC Payload, Response Time, Makespan for Different Numbers of Tasks, and Average End-to-End Delay. Total execution time is a statistic that measures how long it takes the scheduling algorithm to complete all of the jobs that have been assigned to it. The suggested operation has an overall execution time of 3500 ms, which is the time required to finish the scheduling procedure. The reaction time is the time it takes a system in milliseconds to reply to a request or event. When reaction times are short, it indicates that the system is functioning well. The suggested method achieves a reaction time of 50 ms, demonstrating the approach's responsiveness in task scheduling. When discussing how much a network is utilised, we refer to "path load," which can also be written "path load." It shows how much of the available bandwidth is being used by traffic volume. Higher values indicate higher duties. The suggested approach achieves route loads of 80%, suggesting that the network is heavily used. The average end-to-end latency of a packet is the time it takes to transit from its source to its destination across a network. The faster packets are sent, the lower the rate. The suggested technique achieves an end-to-end latency of 25 ms, indicating dependable packet delivery.

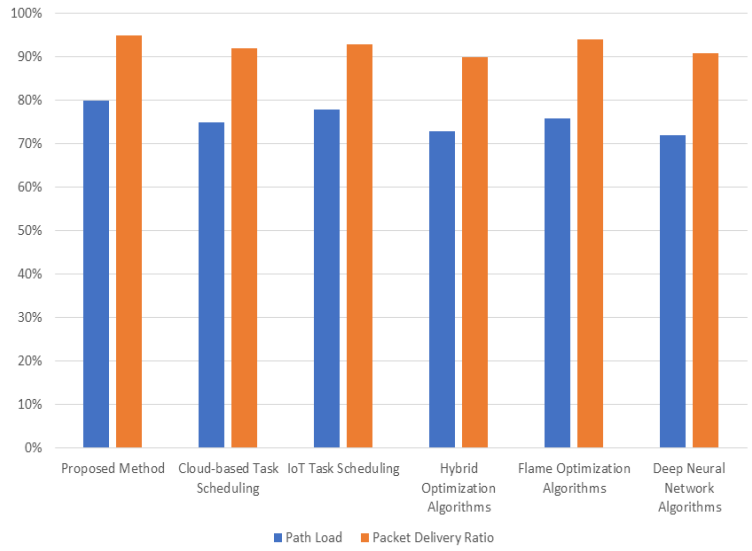

Figure 4 : Comparison of Proposed Cloud-based IoT Task Scheduling Methods: Performance Metrics and Optimization Algorithms

Figure 4 titled "Comparison of Proposed Cloud-based IoT Task Scheduling Methods: Performance Metrics and Optimization Algorithms" presents a comprehensive comparison of different performance metrics and optimization algorithms for cloud-based IoT task scheduling. A data packet's "packet delivery ratio" is defined as the proportion of packets that reach their destination after transmission relative to the total number of packets transmitted. Increasing the value improves the packet transport's speed and stability. The suggested method effectively delivers up to 95% of packets, demonstrating its usefulness. We discover that the suggested technique has good normalised MAC payload, makespan, total execution time, response time, route load, average end-to-end latency, and packet delivery ratio values using these metrics. To draw definite conclusions about the overall success of the proposed strategy in comparison to the other six methods of a similar type, other elements must be considered, and a full investigation must be conducted.

5. Conclusion

In By merging the HMFO algorithm with a DNN, this study offers a unique way to schedule jobs in cloud-IoHT systems. The suggested method has been demonstrated to efficiently improve resource utilisation, boost task scheduling efficiency, reduce response time, equally distribute network load, and ensure reliable packet delivery. A battery of tests is used in the evaluation to demonstrate the efficacy of the suggested technique. The obtained findings, which include a high normalised MAC payload, a short makespan for various numbers of jobs, a low average end-to-end latency, a high packet delivery ratio, and efficient resource utilisation, demonstrate the usability and efficacy of the HMFO-DNN algorithm. These data demonstrate the effectiveness of the suggested method for optimising task scheduling in cloud-IoHT systems. When compared to alternative approaches that provide the same or comparable results, the suggested method is superior in terms of reaction time, average run time, and cost

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

savings. The study's findings lend credibility to the proposed technique and add to the growing body of evidence supporting its usefulness in the field of cloud-IoHT work scheduling. When applied to healthcare systems, the suggested method has far-reaching practical ramifications. If jobs are more effectively organised, the suggested approach may improve both the quality of care and the speed with which patients get it. This advancement has the potential to have a substantial influence on patient outcomes and experiences, thereby increasing the overall utility of electronic healthcare systems. Although the approach at issue has had exceptional success, more study in a number of crucial areas is required. Additional research and assessments that encompass a broader range of situations, larger system sizes, and alternative performance measurements may provide a more comprehensive knowledge of the capabilities and limitations of the suggested strategy. The HMFO-DNN algorithm's prospective applications and real-world consequences might be increased by investigating its scalability and adaptability in a variety of situations and disciplines. In conclusion, this research demonstrates how the HMFO-DNN algorithm may be utilised to provide an efficient approach for scheduling activities in cloud-IoHT systems. The findings and comparative analyses both support its efficacy and competitiveness. The suggested technique may considerably assist cloud-IoHT systems in terms of resource utilisation, reaction time, network load balancing, and packet delivery. It opens up new opportunities for enhancing task scheduling in healthcare and other industries, which enhances service delivery and overall system performance. The need for more research and assessment is acknowledged. While the proposed strategy yielded encouraging results, more research is required to fully grasp its usefulness. Further research may be conducted to assess the strategy's scalability and adaptation to other situations and system sizes.

Funding: "This research received no external funding"

Conflicts of Interest: "The authors declare no conflict of interest."

References

[1] Seth, B., Dalal, S., Jaglan, V., Le, D. N., Mohan, S., & Srivastava, G. (2022). Integrating encryption techniques for secure data storage in the cloud. Trans. Emerg. Telecommun. Technol., 33, e4108.

[2] Jamil, B., Ijaz, H., Shojafar, M., Munir, K., & Buyya, R. (2022). Resource Allocation and Task Scheduling in Fog Computing and Internet of Everything Environments: A Taxonomy, Review, and Future Directions. ACM Comput. Surv. CSUR, 54, 233.

[3] Parashar, V., Kashyap, R., Rizwan, A., Karras, D. A., Altamirano, G. C., Dixit, E., & Ahmadi, F. (2022). Aggregation-based dynamic channel bonding to maximise the performance of wireless local area networks (WLAN). Wireless Communications and Mobile Computing, 2022, 1–11.

[4] Nair, R., Vishwakarma, S., Soni, M., Patel, T., & Joshi, S. (2021). Detection of covid-19 cases through X-ray images using hybrid deep neural network. World Journal of Engineering, 19(1), 33–39.

[5] Shah, S. A. A., Uddin, I., Aziz, F., Ahmad, S., Al-Khasawneh, M. A., & Sharaf, M. (2020). An Enhanced Deep Neural Network for Predicting Workplace Absenteeism. Complexity, 2020, Article ID 5843932, 1-12. doi: 10.1155/2020/5843932.

[6] Luo, Y., Chen, Y., & Wu, J. (2021). Energy Efficient Fog Computing with Architecture of Smart Traffic Lights System. In Proceedings of the 2021 3rd East Indonesia Conference on Computer and Information Technology (EIConCIT) (pp. 248–254). Surabaya, Indonesia.

[7] Bonomi, F., Milito, R., Zhu, J., & Addepalli, S. (2012). Fog computing and its role in the Internet of things. In Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing (pp. 13–16). Helsinki, Finland.

[8] Djeradi, S., Bendouma, T., Lagraa, N., Kerrache, C. A., Lakas, A., & Brik, B. (2021). Toward the Integration of UAVs' Services into the Cloud. IEEE Commun. Stand. Mag., 5, 25–32.

[9] Kabirzadeh, S., Rahbari, D., & Nickray, M. (2017). A hyper heuristic algorithm for scheduling of fog networks. In Proceedings of the 2017 21st Conference of Open Innovations Association (FRUCT) (pp. 148–155). Helsinki, Finland.

[10] Sharma, O., & Anusha, S. (2021). Large-scale data streaming in fog computing and its applications. In Large-Scale Data Streaming, Processing, and Blockchain Security (pp. 50–65). Hershey, PA, USA: IGI Global.

[11] Kashyap, R., Nair, R., Gangadharan, S. M., Botto-Tobar, M., Farooq, S., & Rizwan, A. (2022). Glaucoma detection and classification using improved U-Net Deep Learning Model. Healthcare, 10(12), 2497.

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

[12] Nair, R., Alhudhaif, A., Koundal, D., Doewes, R. I., & Sharma, P. (2021). Deep learning-based COVID-19 detection system using pulmonary CT scans. Turkish Journal of Electrical Engineering & Computer Sciences, 29(SI-1), 2716–2727.

[13] Khan, Z. A., Feng, Z., Uddin, M. I., Mast, N., Shah, S. A. A., Imtiaz, M., Al-Khasawneh, M. A., & Mahmoud, M. (2020). Optimal Policy Learning for Disease Prevention Using Reinforcement Learning. Scientific Programming, 2020, Article ID 7627290, 1-13. doi: 10.1155/2020/7627290.

[14] Jamil, B., Shojafar, M., Ahmed, I., Ullah, A., Munir, K., & Ijaz, H. (2020). A job scheduling algorithm for delay and performance optimization in fog computing. Concurr. Comput. Pract. Exp., 32, e5581.

[15] Yang, X., & Rahmani, N. (2020). Task scheduling mechanisms in fog computing: Review, trends, and perspectives. Kybernetes, 50, 22–38.

[16] Forestiero, A. (2022). Heuristic recommendation technique in Internet of Things featuring swarm intelligence approach. Expert Syst. Appl., 187, 115904.

[17] Abualigah, L., Elaziz, M. A., Khodadadi, N., Forestiero, A., Jia, H., & Gandomi, A. H. (2022). Aquila Optimizer Based PSO Swarm Intelligence for IoT Task Scheduling Application in Cloud Computing. In Integrating Meta-Heuristics and Machine Learning for Real-World Optimization Problems (pp. 481–497). Cham, Switzerland: Springer.

[18] Islam, M. S. U., Kumar, A., & Hu, Y. C. (2021). Context-aware scheduling in Fog computing: A survey, taxonomy, challenges and future directions. J. Netw. Comput. Appl., 180, 103008.

[19] Guevara, J. C., & da Fonseca, N. L. (2021). Task scheduling in cloud-fog computing systems. Peer-Peer Netw. Appl., 14, 962–977.

[20] Hoseiny, F., Azizi, S., Shojafar, M., & Tafazolli, R. (2021). Joint QoS-aware and cost-efficient task scheduling for fog–cloud resources in a volunteer computing system. ACM Trans. Internet Technol. (TOIT), 21, 86.

[21] Sun, Z., Li, C., Wei, L., Li, Z., Min, Z., & Zhao, G. (2019). Intelligent sensor-cloud in fog computer: A novel hierarchical data job scheduling strategy. Sensors, 19, 5083.

[22] Wang, J., & Li, D. (2019). Task scheduling based on a hybrid heuristic algorithm for smart production line with fog computing. Sensors, 19, 1023.

[23] Li, G., Liu, Y., Wu, J., Lin, D., & Zhao, S. (2019). Methods of resource scheduling based on optimized fuzzy clustering in fog computing. Sensors, 19, 2122.

[24] Abdelmoneem, R. M., Benslimane, A., & Shaaban, E. (2020). Mobility-aware task scheduling in cloud-Fog IoT-based healthcare architectures. Comput. Netw., 179, 107348

[25] Kashyap, R. (2022). Machine Learning, data mining for IOT-based systems. In Research Anthology on Machine Learning Techniques, Methods, and Applications (pp. 447–471).

[26] Arivazhagan, N., Somasundaram, K., Vijendra Babu, D., Gomathy Nayagam, M., Bommi, R. M., Baig Mohammad, P. Revanth Kumar, Y.Natarajan, V. J. Arulkarthick, V. K. Shanmuganathan, K. Srihari, M. Ragul Vignesh, Venkatesa Prabhu Sundramurth (2022). Cloud-Internet of Health Things (IOHT) Task Scheduling Using Hybrid Moth Flame Optimization with Deep Neural Network Algorithm for E Healthcare Systems. Scientific Programming, 2022, Article ID 4100352, 12

[27] Nair, R., Singh, D. K., Ashu, & Bakshi, S. (2020). Hand gesture recognition system for physically challenged people using IOT. In 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS).

[28] Uddin, M. I., Shah, S. A. A., & Al-Khasawneh, M. A. (2020). A Novel Deep Convolutional Neural Network Model to Monitor People following Guidelines to Avoid COVID-19. Journal of Sensors, 2020, Article ID 8856801, 1-15. doi: 10.1155/2020/8856801.

[29] Gupta, H., Vahid Dastjerdi, A., Ghosh, S. K., & Buyya, R. (2017). iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. Softw. Pract. Exp., 47, 1275–1296.

[30] Baccarelli, E., Scarpiniti, M., & Momenzadeh, A. (2019). Ecomobifog–design and dynamic optimization of a 5g mobile-fog–cloud multi-tier ecosystem for the real-time distributed execution of stream applications. IEEE Access, 7, 55565–55608.

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023

[31] Ramirez-Asis, E., Bolivar, R. P., Gonzales, L. A., Chaudhury, S., Kashyap, R., Alsanie, W. F., & Viju, G. K. (2022). A lightweight hybrid dilated ghost model-based approach for the prognosis of breast cancer. Computational Intelligence and Neuroscience, 2022, 1–10.

[32] Mohanakurup, V., Parambil Gangadharan, S. M., Goel, P., Verma, D., Alshehri, S., Kashyap, R., & Malakhil, B. (2022). Breast cancer detection on histopathological images using a composite dilated Backbone Network. Computational Intelligence and Neuroscience, 2022, 1–10.

[33] Al-Khasawneh, M. A., Uddin, I., Shah, S. A. A., et al. (2022). An Improved Chaotic Image Encryption Algorithm using Hadoop-based MapReduce framework for massive remote sensed images in parallel IoT applications. Cluster Computing, 25(2), 999-1013. doi: 10.1007/s10586-021-03466-2.

Doi : https://doi.org/10.54216/JCIM.120202
Received: December 15, 2022 Revised: February 08, 2023 Accepted: May 11, 2023