International Journal of Data Science and Artificial Intelligence (IJDSAI)
Volume 3, Issue 5, September – October (2025)

RESEARCH ARTICLE

HMPOA: HYBRID METAHEURISTIC SCHEDULING MODEL FOR OPTIMIZED LOAD BALANCING IN IOT-CLOUD SYSTEMS

Faeiz Alserhani [1,*] and Amjad Alsirhani [2]

[1] Department of Computer Engineering and Networks, College of Computer and Information Sciences, Jouf University, Sakaka, 72388, Saudi Arabia

[2] Department of Software Engineering, College of Computer and Information Sciences, Jouf University, Al Jouf, 72388, Saudi Arabia

[*]*Corresponding e-mail: fmserhani@ju.edu.sa*

Abstract – The massive growth of IoT-based data has increased the necessity to schedule the tasks effectively and distribute the resources in the cloud environment efficiently. Conventional cloud load-balancing approaches are usually incapable of handling diverse workloads, ensuring high resource underutilization, and reducing delays in dynamic operating environments. To overcome such obstacles, the paper proposes a Hybrid Momentum-Pyramid Optimization Algorithm (HMPOA) based IoT-Cloud Scheduling Model that combines the features of Momentum Search Algorithm (MSA) and Giza Pyramid Construction Algorithm (GPCA). The main objective is to efficiently schedule and allocate IoT tasks in a cloud environment. The proposed method maximizes trust, delay and distance metrics to determine optimal execution paths and assign tasks evenly across the virtual machines. The performance of the proposed method is evaluated in terms of higher resource utilization, throughput, energy efficiency, and makespan compared to recent methods. The numerical results show that, at 30 VMs, the proposed method has 84.77% utilization, 5-12% more than APOA, Meta-RHDC, and HDWOA-LBM. Similarly, at 60VMs, the proposed method achieves high scalability 86.90%, compared to APOA of 5.43%, and HDWOA-LBM of 11.8%.

***Keywords** – Load Balancing, Task Scheduling, Internet of Things, Cloud.*

1. INTRODUCTION

Internet of Things devices have significantly influenced the information and communication technology sector [1]. The number of devices linked is projected to reach 30 billion by 2027, which proves the swift expansion of the IoT market. This expansion has contributed to proliferation of IoT applications that generate huge volumes of data with high latency needs. The International Data Corporation (IDC) suggests that by the year 2025, the volume of data generated by IoT devices will be 291 ZB. Cloud computing is regarded as one of the solutions that may effectively manage and store the vast amounts of data produced by Internet of Things devices due to its vast processing and storage abilities [2].

Cloud computing contributes significantly to the successful execution of both scientific and business processes in various industries. It offers scalability and flexibility in pay-per-use solutions that maximize the use of resources in varied applications. There are three broad categories of cloud providers that include Software as a Service (SaaS), Infrastructure as a Service (IaaS) and Platform as a Service (PaaS).The growing user and workloads require more effective ways of load balancing and allocation of resources by cloud systems [3-4]. Although such services as networking, storage and cloud computing may be used with the same purpose, they vary in non-functional parameters that are often referred to as QoS (Quality of Service).

Due to the increasing complexity of cloud systems, load balancing and efficient resource administration is emerging as a significant issue. Many scholars have studied heuristic algorithms and machine learning methods, and have focused on single-objective optimization structures. An effective cloud optimization plan should be designed to control the system as a whole with a balanced approach to addressing various objectives, such as cost, energy usage, performance, and stability [5-6]. A dynamic load balancing algorithm also maximises optimal utilisation of cloud resources, minimises makespan and enhances cloud environment elasticity to accelerate application run time.

A major problem with cloud computing is that it combines several, frequently incompatible optimization objectives. They need to optimize the usage of resources in order to enhance system efficiency. However, it might bring about high operating costs or slowness in responding [7]. Alternatively, to reduce response times, one may need to invest more in some tasks. It may result in high energy usage and low resource effectiveness. The major problem is to come up with optimization strategy which will generate optimal performance through balancing conflicting goals without compromising any other goal in any way [9]. To address these challenges, a novel Hybrid Momentum-

Pyramid Optimization Algorithm (HMPOA) based IoT-Cloud Scheduling Model has been proposed for efficient task scheduling. The paper's main contributions are listed below.

- The primary goal of the research is to develop a HMPOA based IoT-Cloud Scheduling Model to efficiently schedule and allocate IoT tasks in a cloud environment.
- The proposed method integrates Momentum Search and Giza Pyramid Construction methods to provide more accurate, trust, and delay-optimized task assignment in the cloud.
- The performance of the proposed method is evaluated in terms of higher resource utilization, throughput, energy efficiency, and makespan compared to recent methods.

The remaining part of the paper is going to be organized as follows: Section 2 will be the literature survey in detail, Section 3 will be a description of the proposed HMPOA-based IoT-Cloud Scheduling Model, Section 4 will be the results and discussion of the proposed framework, and Section 5 will be the conclusion and future work.

2. LITERATURE SURVEY

The concept of load balancing in cloud computing has changed over time as the workloads of heterogeneous aspects become more complex and the need to employ smart scheduling techniques increases. The recent research has laid a lot of emphasis on metaheuristics, hybrid optimization and intelligent controllers to obtain scalable dynamic load allocation.

In 2023 Al Reshan, M.S., et al., [9] suggested a global optimization and rapid convergence solution to load balancing in cloud computing. The GWO-PSO combined approach, which optimizes the benefits of global optimization and fast convergence for load balancing, is heavily emphasized in the suggested strategy. The best result of the proposed GWO-PSO algorithm objective function is an enhancement of PSO up to 97.253% in convergence and the total time response of the proposed approach is 12% lower.

In 2023 Ramya, K. and Ayothi, S., [10] suggested a Hybrid dingo and whale optimization algorithms based optimal load balancing (HDWOA-LBM) to the cloud computing environment. Through the simulation of dingo hunting behaviors, which are analogous to tasks, the HDWOA-LBM method is proposed to find the optimal way to assign incoming tasks to the appropriate virtual machine (VM). Simulation tests of the proposed HDWOA-LBM method show the following results as the throughput of 21.28%, the highest reliability of 25.42, the lowest makespan of 22.98, and the best resource allocation of 20.86% of the intelligent load balancing are achieved.

In 2024 Ghafir, S., et al., [11] suggested Intelligent PSO-based feedback controller to efficiently perform Load balancing in cloud computing. This multi-objective algorithm aims to achieve optimal bilateral transposed convolution filtering, low response time, scalability, throughput, and high service quality. In order to maintain service level agreement with the cloud, a Double Deep Q proximal model with a feedback controller is recommended. The conditional GAN feedback controller eliminates the possibility of a single point of failure.

In 2024 Singhal, S., et al., [12] proposed a metaheuristic-based cloud computing solution, a Rock Hyrax-based load balancing algorithm overcomes local maxima and efficiency of power. It lowers makespan and energy usage in data centers by 10-15 percent and total energy usage by 8-13 percent, proving to be effective in enhancing the performance of systems and improving the allocation of resources. This algorithm is based on metaheuristics and offers a more reliable and efficient method of allocating resources in cloud computing.

In 2024 Khaleel, M.I., [13] proposed Region-aware dynamic job scheduling and resource efficiency in cloud computing environments in the context of load balancing. A novel coalitional game-theoretic process based on merge-and-split is used to group nodes into clusters, and an enhanced Sparrow Search Algorithm (ISSA) is used to get around the slow convergence and local optimization. In contrast, the Latency overhead is decreased by 9%, the Processing time is decreased by 14%, the workload imbalance is decreased by 15%, the energy consumption is decreased by 19%, the idle periods are decreased by 26%, throughput increases by 32%, resource availability increases by 22%, and resource efficiency increases by 27%.

In 2025 Krishna, M.S.R. and Vali, D.K., [14] suggested Cloud computing load balancing that is dynamic and powered by Meta reinforcement learning through a hybrid Lyrebird Falcon optimization (Meta-RHDC). By predicting their loads using convolutional and recurrent neural networks, it divides virtual machines into groups that are overloaded and underloaded. In comparison to alternative methods, this approach significantly improves load balancing and task scheduling. CloudSim platform tests show significant improvements in key performance metrics, making Meta-RHDC a leading choice for dynamic load balancing in cloud computing.

In 2025 Hegde, S.K., et al., [15] suggested a Hybrid Adam_Pufferfish Optimization Algorithm (AdamPOA) Based Load Balancing in Cloud Computing. Cloud computing (CC) provides solutions to scheduling, security, and load balancing, especially in virtual machines (VMs). A helpful model that has been designed to work with LB in CC is the Adam_Pufferfish Optimization Algorithm (AdamPOA). It allocates tasks to virtual machines (VMs) with Deep Fuzzy Clustering, optimizes cloud constraints with hybrid AdamPOA, and allocates tasks by priority. AdamPOA meets load, reliability, capacity, and resource availability of 0.880, 0.915, and 0.857 respectively.

The current approaches have optimized only a few parameters and are unable to retain accuracy when workloads are dynamic and heterogeneous. Numerous methods encounter such problems as slow convergence and local minima, which minimizes the predictability of schedules. Some of the techniques are not scalable and energy efficient resulting in increased overhead and uneven resource usage, to overcome these drawbacks, a novel HMPOA based IoT-

Cloud Scheduling Model has been proposed to efficiently schedule tasks in a cloud environment

3. PROPOSED METHODOLOGY

In this section, a novel HMPOA based IoT-Cloud Scheduling Model has been proposed to efficiently schedule and allocate IoT tasks in a cloud environment. Initially, Cloud users send the data and tasks produced by IoT devices to the cloud. The System Accessibility Layer authenticates and validates a user request and translates the tasks into cloudlet form. The Hybrid Momentum-Pyramid Optimization Algorithm (HMPOA) is applied to determines the optimal distribution of resources to each task. This optimized plan is used by a Resource Manager to allocate tasks to respective Virtual Machines (VMs) in the Virtualization Layer. These VMs are deployed in the physical infrastructure of the cloud, which is made up of Physical Machines (PMs). The overall IoT-Cloud HMPOA Architecture is depicted in the figure 1.

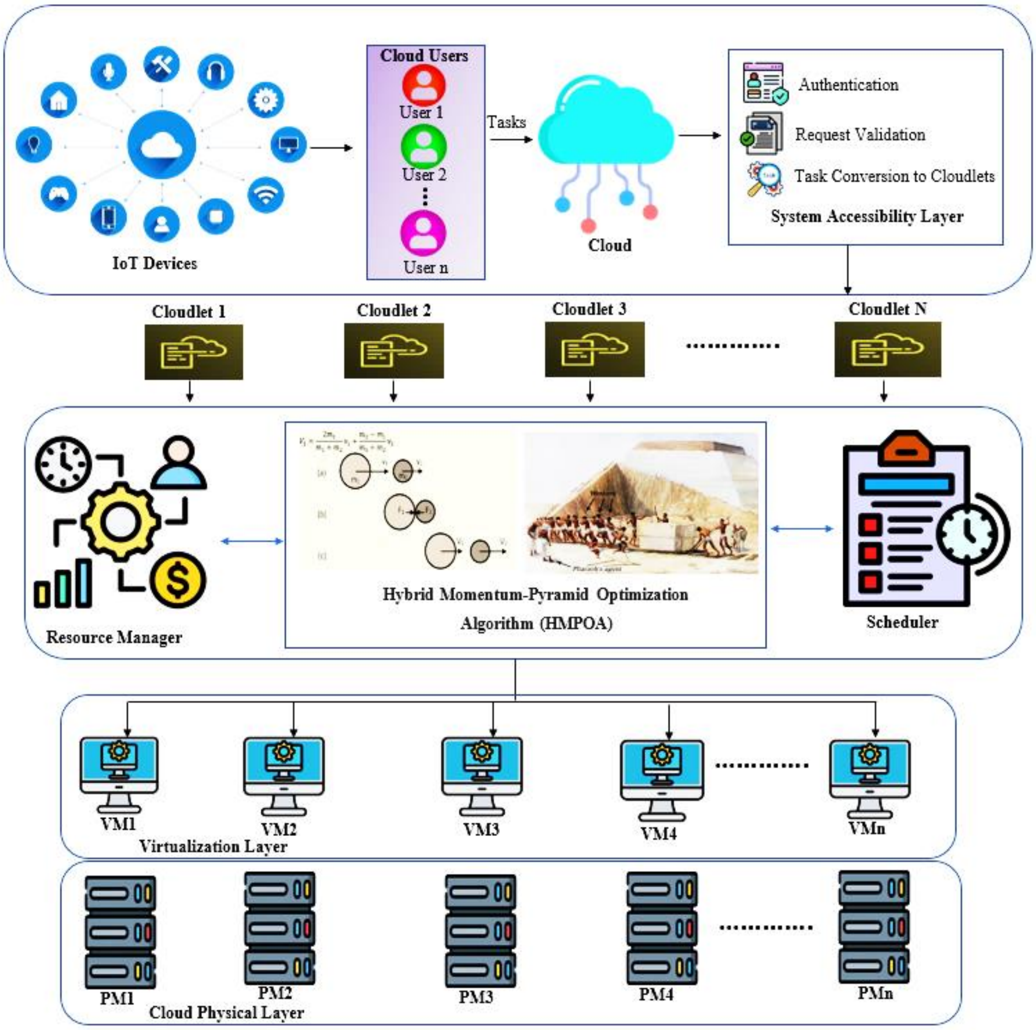

Figure 1. IoT-Cloud HMPOA Architecture

3.1 IoT Devices and Cloud User

IoT devices and cloud users serve as data sources generating and submitting tasks that require processing. These tasks may include sensor readings, user requests, or real-time application data. The input to the system is the set of tasks which are sent to the cloud for resource allocation and load-balanced execution.

3.2 System Accessibility Layer

The system accessibility layer is where the cloud users can easily submit their cloudlets to be processed within the cloud environment. This layer provides secure and efficient communication, isolates the underlying complexity of the cloud, and provides authentication, verification of requests, data formatting and transfers the tasks to the resource manager.

Cloud Resource Manager

The cloud resource manager is also used to communicate in order to offer the availability and computing capacity of VMs to the HMPOA scheduler. The proposed HMPOA is deployed over the virtualization to effectively utilize VMs in a load-balanced fashion. This algorithm gives a balance distribution of cloudlets among the cloud system VMs.

3.3 Scheduler

The HMPOA is a scheduling protocol that is developed to optimize the allocation of tasks in heterogeneous VMs within cloud computing systems. The algorithm helps to overcome the crucial problem of effective resource usage, equal distribution of loads, and minimized execution times. Using the compute ability of VMs and dynamically changing the allocation of tasks based on workload fluctuations, HMPOA is able to assure that tasks are allocated with the least amount of overhead used in execution. The initial phase makes every VM have a portion of work that matches its strengths to facilitate fair distribution of resources.

3.3.1 Hybrid Momentum-Pyramid Optimization Algorithm (HMPOA)

The Momentum Search Algorithm (MSA) is based on the laws of Newton and especially the concept of momentum conservation. Under this method, every candidate solution will be considered a mass-based object and their interactions will be of momentum-preserving dynamics, to lead the search to optimal solutions. During each of the iterations, an external body crashes itself on all solution bodies and moves towards the best outcome. In MSA, optimum placement is achieved by letting the external guiding body drag the solution bodies to more favourable areas in the search space. The positions of all the bodies are dynamically updated by means of the functions $De(t)$ and $Dist(t)$, which govern the attraction and dispersion with time. Equally, Giza Pyramids Construction Algorithm (GPCA) is a meta-heuristic algorithm that is based on pyramid-building strategy in the past, as the systematic steps or stages lead the search and exploitation of potential solutions. Just like Giza Necropolis, the GPCA is a site that comprises of giant three pyramids; they are all built under the fourth dynasty of ancient Egypt. GPCA production is received with the help of pushing of the stone slabs on the slope and labors. In such a way, the location of one body of labors substitutes another. This option alters the motion and strength ratio of the stone slab. Some of the labors can also be potentially replaced during the building process and moved elsewhere. To maximize the $Trus(t)$ the updating location of labors in GPCA is used. The algorithmic process of hyb MSA-GPCA is described in the following. The momentum search population and Giza Pyramids Construction population are being initialized evenly in the solution space according to the equation below (1),

$$Y_j, g_k(t'') = \left(y_j^c(t'') + \eta_k nh\, cos\theta\right) \tag{1}$$

Where, Yj, $(t")$ refers to the $j\ th$ solution body at time $t"$. The gk symbol is used to denote the force of kinetic-resistance as introduced in the GPCA model. Parameters m and c show the dimensional settings of MSA water bodies. In the GPCA expression, n is the mass of the stone slab, k is a coefficient of kinetic-resistance and h is the earth load, which also indicates the slope angle of the construction ramp.

GPCA and MSA input parameters are randomly generated after the process of initialization. In this case, there is the maximum fitness values and selection of the superior path depends on a fitness function. Produce the random number of solutions based on the initial values. The trust, delay and distance between the node are connected to the fitness of the identical functionality. Equation (2) is the calculation of the fitness.

$$Fitness_{taskschedulex} = \left\{\left[1 + \frac{Del(t)}{Del_{norm}}\right] + \left[1 - \frac{Dis(t)}{X^{n}{*B*M}}\right] + Resource(t)\right\} \tag{2}$$

From equation (2) $Di(t)$ is the distance between nodes, $Del(t)$ is the delay, $Trust(t)$ are the trust. Fitness solutions need to maximize trust, which also minimizes distance and delay. Therefore, GPCA optimizes the $Trus(t)$, $Del(t)$ and $Dis(t)$ the parameter by MSA to choose the optimal path. The important concept of GPCA is that labours cause the moving stone slab to move forward in a continuous manner in order to advance the capabilities of control and feasible lead of the stone slab. These shock waves trigger the process of attaining non-repetitive displacement on the employee through forcing the stone slab well. Therefore, it is possible to compute the motion of the rock on the slope as shown in equation (3),

$$Displacement\ of\ stoneslap = \frac{u_o^2}{2h(sin\theta + \eta_k cos\theta)} \tag{3}$$

Where h is the gravity of the earth, uo is the original velocity of the stone slab. The above sentence is employed to control the new status of the employees. Therefore, the moment where the workers exert force to move the stone slab can be defined as written in equation (4).

$$Movement\ of\ workers = \frac{u_o^2}{2h\ sin\theta} \tag{4}$$

In where, θ represents the ramp angle. After the changes in the movement (trust path at lower power level) of the stone slab and the change in position (trust path selection) of the workers are established on equation (4) were computed, a new location could be obtained by that of the above equations after the above equations were subtracted. The new location of the stone slab is calculated by summing the former location of the stone slab and its displacement, which depends on the movement of the workers. This new calculated position is recorded as this adjusted point. Therefore, the renewed poses of the stone slab and the workers are structured and stated in Equation (5) below.

$$\vec{Q} = \left(\vec{Q}_i + c\right) \times y\vec{\sigma}_i \tag{5}$$

Where, $Q^{\to}\ i$ denotes the present location, c is the motion of the stone slab, y represents the shift of labour, $\sigma\ i$ represents the random variable which is described by the Uniform and Normal distribution. This employee movement is used to analyze the best solution that can be used now to achieve the best global solution. This movement of workers is employed with the aim of determining the most appropriate solution that exists so that they can arrive at the most appropriate global solution. This means that the trusted node will be acquired and the deletion of the untrustworthy node. This flow of slabs of stones is carried out to choose a trusted node with low power values. The new position will maximize the $Trust(t)$, as in equation (5). In each repetition all the solution bodies are fixed and the separate exterior body is indicated in the space denoted exterior body. When an interaction between one body and another takes place, it is diverted to a more ideal position. This exchange of momentum leads the external bodies to slow down as well as their mass becoming minimal, and the total mass acting on them attaining its peak. The equations (6) and (7) give the equivalents of computing the weight and velocity of these external bodies respectively.

$$N(t'') = 1 - \frac{t''-1}{T''-1} \tag{6}$$

$$V^{(c)}j(t'') = S_1.\left(1 - \frac{t''-1}{T''-1}\right).V_{best\ j_{max}}^{c\ \ \ c} \tag{7}$$

where $(t")$ is the mass solution body at time $t"$, $T"$ represents the largest number of iterations. Then $V\ (c)\ j\ (t")$ is the speed of the exterior body where c is the dimension, and $j\ th$ is the system body iteration 1 s is the random term, $Vmax$ is the reality of the random number, $ybest\ c\ (t")\ and\ yj\ c\ (t")$ and the values of system body c dimensions are the best fitness of $j\ th$ system body iteration. Following the collisions, the new position is found using speed equations (7), This demonstrates that the current status of the respective body is defined by the ratio between its past location and its after collision velocity. Therefore, the new position of the body is presented as follows in Equation (8),

$$y_j^c(t''+1) = y_j^{(c)}(t'') + S_2 U_j^c(t'') \tag{8}$$

The equation above $Uj\ c\ (t")$ represents the updating of position of system speed with c dimension and $j\ th$ system body iteration at $t"$ one of the two values and $s2$ is randomly distributed unchanged term in the range of [0, 1]. Using equation (8), the new position will be the one that minimizes the $De(t)\ and\ Dis(t)$. The process is applied repeatedly in Equations (6) to (7) until the desired requirements are met at which the process is stopped once the optimal solution is found. Optimal path at minimum distance and minimum delay will be given as final equations. The hyb - MSA-GPC algorithm output provides an optimal path which continues to reiterate step 3 until halting criteria = + 1. The best of the ways, trust, transmits the information to BS.

3.4 Cloud Physical Layer

The physical and virtual layers provide Cloud users with Infrastructure-as-a-Service (IaaS) and Platform-as-a-Service (PaaS). The computing, memory, and storage hardware that comprise the physical machines (PMs) of the total computing capacity of the datacenter constitute the physical layer. The machines are commonly referred to as PM1, PM2, ... PMN.

3.5 Virtualization Layer

Virtualization layer provides computing and storage capabilities to the cloud users in a transparent mode by dynamically assigning these capabilities to the virtual machines (VMs). The cloud resource manager is located above this layer, and its main responsibility is to manage the lifecycle of VMs such as their creation, deletion, and movement whenever the computing needs of the system necessitate such activities.

4. RESULT AND DISCUSSION

This part will show the cloud computing model setup and also defines the datasets that were used in the experimental analysis. The software is run on an Intel CoreTM i5- 4030U 1.90 GHz 2.49 GHz 12 GB of RAM computing machine. HMPOA is evaluated through its performance in simulation, with the Cloud Simplus simulator. Experimental assessment is conducted on 17 to 60 virtual machines that are deployed on 40 cloud host systems. The different instances of the GoCJ dataset are run by way of the VM configurations, which have their MIPS (Millions of Instructions Per Second) ratings. GoCJ dataset covers a great variety of job types: small, medium, large, extra-large and massive jobs. These jobs are arranged and classified in the dataset in accordance with their sizes and their computation complexities.

4.1 Performance Metrics

The proposed model is compared with the existing techniques such as HDWOA-LBM, Meta-RHDC, and APOA in terms of resource utilization, throughput and energy consumption.

Resource Utilization: It shows the efficiency with which the cloud resources (CPU, memory, VMs) are utilized when performing a task.

$$RU(\%) = \left(\frac{Total\ used\ resources}{total\ available\ resources}\right) \times 100\%$$

Throughput: This is the amount of tasks that the system can complete in a time period.

$$T = \frac{No.of\ tasks\ executed\ successfully}{total\ execution\ time}$$

Energy Consumption: It is the overall energy that is consumed by cloud physical machine and virtual machines to perform tasks.

$$EC = \sum_{i=1}^{n} P_i \times T_i$$

4.2 Comparative Analysis

Makespan is a performance measure used in scheduling, job management that is the sum of time necessary to complete a set of tasks, beginning with the start of the initial task and ending with the completion of the final task. It is useful in quantifying the overall productivity of a system in terms of workload management. The importance of minimizing makespan is as follows: it leads to utilization of resources optimally, as idle times are minimized, the entire system throughput is maximized, tasks are fulfilled faster and user satisfaction increases due to the fact that the service level agreements (SLAs) were met or surpassed.

Table 1. Resource Utilization with 30 VMs

Number of Tasks	HDWOA-LBM [10] (%)	Meta-RHDC [14] (%)	APOA [15] (%)	Proposed (%)
1000	64.21	68.32	71.45	75.88
2000	66.87	70.55	74.29	78.64
3000	69.13	73.02	76.45	80.91
4000	70.54	74.88	78.23	82.30
5000	72.01	76.31	79.60	84.77

Table 2. Resource Utilization with 45 VMs

Number of Tasks	HDWOA-LBM [10] (%)	Meta-RHDC [14] (%)	APOA [15] (%)	Proposed (%)
1200	65.88	69.45	72.80	76.93
2200	68.12	72.03	75.56	79.28
3200	70.44	74.67	77.94	81.62
4200	72.31	76.11	79.41	83.20
5200	73.90	77.69	80.63	85.47

Table 3. Resource Utilization with 60 VMs

Number of Tasks	HDWOA-LBM [10] (%)	Meta-RHDC [14] (%)	APOA [15] (%)	Proposed (%)
1500	66.34	70.02	73.50	77.45
2500	68.91	72.58	76.00	79.88
3500	71.55	75.13	78.32	82.25
4500	73.42	76.95	80.13	84.66
5500	75.10	78.80	81.47	86.90

Resource utilization reflects the effectiveness with which available computational resources are consumed. It is typically calculated by comparing the amount of resources in use to the overall capacity of the system. Achieving higher utilization promotes better workload distribution, minimizes idle resources, and improves overall system performance by reducing potential bottlenecks.

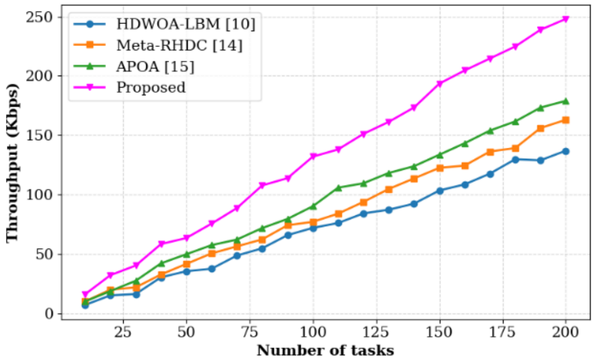

Figure 2. Throughput Comparison

Figure 2 shows the relationship between the number of tasks and throughput (Kbps) for four scheduling methods such as HDWOA-LBM, Meta-RHDC, APOA, and the Proposed method. The throughput level increases steadily with the amount of tasks in each method, and this shows that they can handle data better when workloads are heavier. The Proposed approach is the highest throughput at all the levels of tasks, with a maximum of almost 250 Kbps at 200 tasks. While, HDWOA-LBM had the lowest throughput across the observed range. In general, the findings provide clear evidence that the Proposed method is better in terms of scalability and efficiency in comparison with current methods.

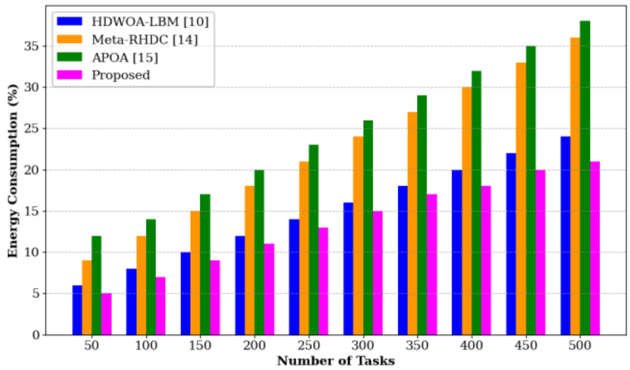

Figure 3. Energy Consumption

Figure 3 shows the comparisons between the four load-balancing strategies, including HDWOA-LBM, Meta-RHDC, APOA, and the Proposed method, in terms of the increasing numbers of tasks. When the number of tasks increases to 500, every method exhibits increased energy consumption because of the increased computational needs. Nevertheless, the Proposed method always shows the minimum energy usage at all task levels, which suggests better resource use. Conversely, APOA and Meta-RHDC have the most energy consumption, particularly after 250 tasks, and less energy efficiency is demonstrated at a higher workload. HDWOA-LBM has a moderate score, yet it consumes more energy than the Proposed model.

5. CONCLUSION

In this paper, a novel HMPOA based IoT-Cloud Scheduling Model has been proposed to efficiently schedule and allocate IoT tasks in a cloud environment. This integrates Momentum Search and Giza Pyramid Construction methods to provide more accurate, trust, and delay-optimized task assignment in the cloud. The performance of the proposed method is evaluated in terms of higher resource utilization, throughput, energy efficiency, and makespan compared to recent methods such as HDWOA-LBM, Meta-RHDC, and APOA. The simulation experiment was done using cloud sim Plus with GoCJ dataset. The numerical results show that, at 30 VMs, the proposed method has 84.77% utilization, 5-12% more than APOA, Meta-RHDC, and HDWOA-LBM. Similarly, at 60VMs, the proposed method achieves high scalability 86.90%, compared to APOA of 5.43%, and HDWOA-LBM of 11.8%. The HMPOA algorithm can have computational overhead in processing very large-scale IoT workloads because of its hybrid form. Thus, future research can aim to create lightweight or parallelized HMPOA versions to decrease complexity and increase responsiveness to massive IoT settings.

CONFLICTS OF INTEREST

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

FUNDING STATEMENT

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors

ACKNOWLEDGEMENTS

The author would like to express his heartfelt gratitude to the supervisor for his guidance and unwavering support during this research for his guidance and support.

REFERENCES

[1] U. K. Lilhore, S. Simaiya, Y. N. Prajapati, A. K. Rai, E. S. Ghith, M. Tlija, T. Lamoudan, and A. A. Abdelhamid, "A multi-objective approach to load balancing in cloud environments integrating ACO and WWO techniques," *Scientific Reports*, vol. 15, no. 1, p. 12036, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[2] M. Ghorbani, N. Khaledian, and S. Moazzami, "ALBLA: an adaptive load balancing approach in edge-cloud networks utilizing learning automata," *Computing*, vol. 107, no. 1, p. 34, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[3] A. R. Khan, "Dynamic load balancing in cloud computing: optimized RL-based clustering with multi-objective optimized task scheduling," *Processes,* vol. 12, no. 3, p. 519, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[4] G. Verma, "Load balancing in cloud environment using pposition based spider monkey optimization," Wireless Personal Communications, vol. 137, no. 2, pp. 977–996, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[5] A. Khunger, "Fault-Tolerant Load Balancing In Cloud-Based Financial Analytics: A Reinforcement Learning Approach," *International Journal of Innovation Studies*, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[6] K. D. S. Devi, D. Sumathi, V. Vignesh, C. Anilkumar, K. Kataraki, and S. Balakrishnan, "CLOUD load balancing for storing the internet of things using deep load balancer with enhanced security," *Measurement: Sensors*, vol. 28, p. 100818, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[7] G. Saranya, G. B. Jebamalar, and C. Santhaiah, "Dynamic load balancing in cloud computing using hybrid Kookaburra-Pelican optimization algorithms," *International Journal of Data Science and Artificial Intelligence*, vol. 2, no. 4, pp. 98–104, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[8] M. Jesi, A. Appathurai, M. Narayanaperumal, and A. Kumar, "Load balancing in cloud computing via mayfly optimization algorithm," *Revue Roumaine des Sciences Techniques – Série Électrotechnique et Énergétique*, vol. 69, no. 1, pp. 79–84, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[9] M. S. Al Reshan, D. Syed, N. Islam, A. Shaikh, M. Hamdi, M. A. Elmagzoub, G. Muhammad, and K. H. Talpur, "A fast converging and globally optimized approach for load balancing in cloud computing," *IEEE Access,* vol. 11, pp. 11390–11404, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[10] K. Ramya and S. Ayothi, "Hybrid dingo and whale optimization algorithm-based optimal load balancing for cloud computing environment," *Transactions on Emerging Telecommunications Technologies*, vol. 34, no. 5, p. e4760, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[11] S. Ghafir, M. A. Alam, F. Siddiqui, and S. Naaz, "Load balancing in cloud computing via intelligent PSO-based feedback controller," *Sustainable Computing: Informatics and Systems*, vol. 41, p. 100948, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[12] S. Singhal et al., "Energy efficient load balancing algorithm for cloud computing using rock hyrax optimization," *IEEE Access*, vol. 12, pp. 48737–48749, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[13] M. I. Khaleel, "Region-aware dynamic job scheduling and resource efficiency for load balancing based on adaptive chaotic sparrow search optimization and coalitional game in cloud computing environments," *Journal of Network and Computer Applications*, vol. 221, p. 103788, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[14] M. S. R. Krishna and D. K. Vali, "Meta-RHDC: Meta reinforcement learning driven hybrid lyrebird Falcon optimization for dynamic load balancing in cloud computing," *IEEE Access*, vol. 13, pp. 36550–36574, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[15] S. K. Hegde, R. Hegde, C. N. Kumar, R. Meenakshi, R. Raman, and G. M. Jayaseelan, "Hybrid Adam_POA: Hybrid Adam_Pufferfish optimization algorithm based load balancing in cloud computing," *SN Computer Science*, vol. 6, no. 2, p. 178, 2025. [CrossRef] [Google Scholar] [Publisher Link]

AUTHORS

Faeiz Alserhani, Received his Bachelor's degree in Computer Engineering from King Saud University, Riyadh, SA, an M.S degree in Computer and Information Networks from the University of Essex, UK, and a Ph.D. degree in Network and Information Security from the University of Bradford, UK. He is a professor assistant at the Department of Computer Engineering and Networks at Jouf University, SA. His interests cover several aspects across Network Security, cyber security, Intrusion Detection Systems, and the Application of AI in Cybersecurity

Amjad Alsirhani is an Associate Professor at the Faculty of Computer Science, Jouf University, where he serves as the Head of the Software Engineering Department. He earned his bachelor's degree in computer science from Jouf University, followed by advanced studies at Dalhousie University in Canada, where he was awarded both a Master of Computer Science (M.C.S.) in 2015 and a Ph.D. in 2020. His academic journey has been marked by a strong commitment to both research and teaching. In addition to his role at Jouf University, Dr. ALSIRHANI holds an Adjunct Professor position at Dalhousie University, maintaining an active academic collaboration between the two institutions. Dr. ALSIRHANI's research interests span a wide array of topics in the field of computing, including Cybersecurity, Network Security, Cloud Computing Security, Distributed Computing Systems, as well as Machine Learning and Deep Learning. His work explores innovative approaches to enhance data protection and system resilience in these fields.

Arrived: 24.09.2025
Accepted: 30.10.2025