scientific reports

OPEN

Improved multi-strategy secretary bird optimization for efficient IoT task scheduling in fog cloud computing

K. Sangeetha[1]✉ & M. Kanthimathi[2]

Applications designed for real-time IoT operations improve cloud-based service utilization due to their rapid scalability. Though cloud computing appears to be more effective for data processing and storage in a range of IoT applications, its real-time scalability presents issues in fulfilling the demands of network bandwidth and latency-sensitive applications. In this context, fog computing is shown to be a complementary paradigm to cloud computing, providing extra benefits and capabilities aimed at extending cloud services to end users and edge devices. Due to the restricted capabilities of fog nodes, only lightweight activities can be conducted locally, while jobs requiring more processing time are handled in the cloud. As a result, an Improved Multi-Strategy Enhanced Secretary Bird Optimization Algorithm using Reinforcement Learning (IMSESBOA + RL) for IoT Task Scheduling (TS) mechanism is presented to reduce data processing time and enhance Quality of Service (QoS) in fog-cloud computing. This IMSESBOA + RL approach is designed as an efficient scheduling model that investigates and processes various scalable quantities of tasks while minimizing latency and energy costs. It used a multi-objective methodology based on Secretary Bird Optimization Algorithm's (SBOA) balanced exploration and exploitation capabilities, which has multi-strategy benefits in terms of maximizing resource consumption rate and shortening makespan. It further uses RL for dynamically adapting to the new workloads by excelling in learning optimal strategies using the interaction of trial and error with the environment. The simulation findings of the IMSESBOA + RL approach verified that it reduced makespan by 19.42% and execution time by 18.32% compared to the baseline approaches with various jobs originating from IoT applications.

Keywords Internet of things (IoT) applications, Task scheduling (TS), Secretary bird optimization algorithm (SBOA), Fog-Cloud computing, Quality of service (QoS)

In recent past, technology of Internet of Things (IoT) has evolved as an indispensable paradigm which helped in enhancing diversified dimensions of human life for facilitating comfort in their day-to-day activities[1]. A myriad of devices such as classical smart devices, wearable devices, machines and sensors are connected through the Internet for the objective of interconnecting the benefits of IoT devices and their associated technologies such that support the actions carried out by humans[2]. These interconnected devices facilitated different numbers of services that are related to the domains of vehicular networking, logistics, smart retails, intelligent traffic control, health monitoring, and so on. But the amount of data produced by these smart devices needs to be processed potentially for the purpose of extracting essential amounts of information that assists the applications of IoT in a more reactive manner[3]. But these amounts of data generated in the monitoring environment cannot be significantly processed by IoT devices since they possess limitations in terms of storage and processing capabilities. At the same time, cloud computing represents a distributed computing paradigm which wide opens the option of providing an on-demand pool of resources to users who are connected over the Internet[4]. The tasks should be assigned to the more powerful cloud nodes for the purpose of utilizing the rich number of available resources possessed by them since the IoT devices are prone to the inherent limitations of storage, low computing power and restricted battery life[5]. But the exponentially growing IoT devices with the support of the Internet increase the possibility of generating unprecedented amounts of data which could not be reliably

[1]Department of Electronics and Communication Engineering, Sri Sai Ram Institute of Technology, Chennai 600044, Tamil Nadu, India. [2]Department of Computer Science and Engineering in Internet of Things (IoT), Sri SaiRam Engineering College, Chennai 600044, Tamil Nadu, India. ✉email: ksangeethaece@gmail.com

handled by cloud datacenters which cannot satisfy the requirement of IoT applications in dynamic manner[6,39]. However, the possibility of realizing the influences of transmission delay and network congestion is unavoidable and they considerably degrade the performance of QoS when deadline-driven IoT applications associated with the applications of online gaming, smart transportation, smart cities and healthcare, and so on need to be executed with delay sensitivity in real-time[7,40]. These limitations realized in the cloud computing paradigm motivated the option of extending cloud resources to network edge such that the IoT tasks associated with these applications can be executed within the considered deadline in real time.

To extend benefits of cloud computing paradigm to network edge, Cisco introduced fog computing for minimizing the cloud datacenters' burden imposed during the process of executing the IoT tasks[8]. This fog computing paradigm comprises of different types of devices that include surveillance cameras, routers, switches, gateways, access points, embedded servers, cellular base stations and controllers. This fog computing paradigm facilitates the benefits of low latency during the execution of delay-sensitive IoT applications. Attaining location awareness, geographical distribution of IoT devices, supporting mobility, minimizing the rate of energy consumption and conserving the degree of network bandwidth[9]. However, the capabilities of fog nodes based on storage and computing capabilities are insufficient during the execution of large IoT applications like the one that pertains to big data analytics. Moreover, cloud and fog nodes need to cooperate with one another such that they can support computation-intensive and delay sensitive IoT tasks with the development of a paradigm termed cloud-fog computing which exploits benefits of cloud and fog resources simultaneously[10]. In specific, the widely utilized cloud-fog computing architecture used for achieving the benefits of cloud-fog computing is the three-tier architecture depicted in Fig. 1. This three-tier architecture is used for implementing cloud-fog computing comprises of 3 layers that pertain to IoT devices, fog nodes and cloud datacenters[11]. The bottom-most tier includes several numbers of IoT devices which incorporated GPS capability for enabling the users to submit the required tasks depending on the on-demand increased in the network. The middle tier represents the fog computing platform for varying number of edge nodes which play an anchor role in computing and handling the data which are very closer to the place where it is getting generated[12]. The final tier represents the cloud computing platform which opens the option of providing a massive number of resources as datacenters. It further facilitates a convenient environment which offers an extensive range of computing resources. These cloud-fog systems employed the technology of virtualization for providing resources in the form of VMs associated with the fog and cloud nodes[13].

Moreover, this cloud-fog system prevents the heterogeneity of server for enhancing the rates of resource utilization and achieving server consolidation. Moreover, the system of virtualized cloud–fog is determined to be efficient and effective in deploying the bag-of-tasks applications that are related to video encoding, video decoding computational biology and massive searches. These applications comprise of several independent tasks which possess the capability of executing them in parallel[14]. In addition, Task Scheduling (TS) is a major challenge in cloud-fog system for achieving the requirements of the IoT applications.

In cloud-fog computing paradigm, TS algorithm attempts to identify the best task assignments to the available VMs with the view to satisfying the necessitated objectives of scheduling[15]. This TS process needs to be achieved by satisfying the deadline requirements of real-time tasks with reduced costs of execution, makespan values, optimized energy consumptions and maximized resource providers' profits. These TS algorithms may be either static or dynamic based on the consideration or ignoration of the deadline constraints during the process of determining optimal schedules[16]. In specific, static TS approaches are more ideal for handling the execution of moderate and small sized IoT tasks. But static TS algorithms are not completely ideal for dynamic IoT environments since they generate large-sized application tasks with different arrival times into the

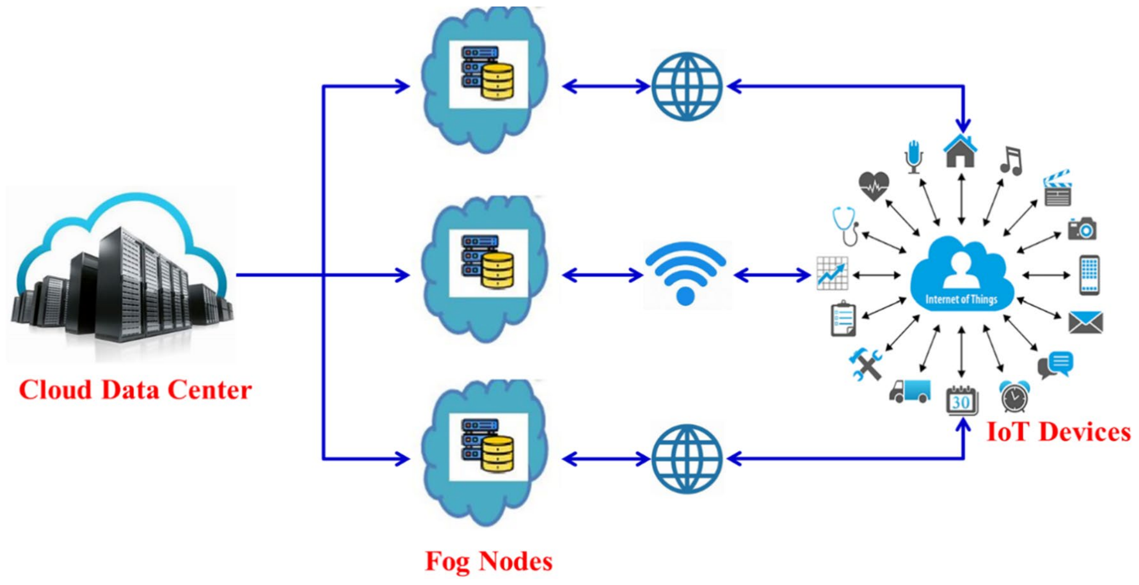


Fig. 1. Three-tier architecture of cloud-fog computing paradigm

implementation environment. This process of dynamic TS on cloud-fog computing scenario is an optimization problem with NP-Hard complexity. In this context, metaheuristic schemes are seen to be ideal solutions for addressing the process of TS which incurs NP-Hard complexity[17]. Hence, it is identified that the development of dynamic real-time TS mechanism is highly required for executing the task in delay-sensitive IoT applications. The metaheuristic algorithms used for attaining TS process in cloud-fog computing domain contributed to literature includes Ant Colony Optimization (ACO), Genetic Algorithm (GA), Artificial Bee Colony (ABC), Particle Swarm Optimization (PSO), moth flame, bees' life and so on[18].

Motivation

The IoT applications in general generate large amounts of data that demand processing, storage and analysis for the objective of determining potential judgements that satisfy the goals and requirements of the users. In cloud-fog computing paradigms, metaheuristic optimization algorithms-based dynamic TS approaches are significant for satisfying user demands within the necessitated response time as it is essential for executing the IoT tasks without delay in real time[19]. In specific, metaheuristic algorithms are utilized for determining efficient solutions that helps in solving the engineering and optimization problems in real-world through the inclusion of nature inspired behaviors. The bio-inspired metaheuristic optimization algorithms are usually developed by mimicking the behaviors of animals, ants, birds, fishes, termites and so on. These behaviors considered in the development of bio-inspired metaheuristic optimization algorithms may be associated with the social organization, swarming, foraging, hunting, reproduction, mating of the species, and so on. These behaviors of the species are analogical to exploration and exploitation stages of bio-inspired metaheuristic optimization algorithms that are utilized for attaining the objectives of the real engineering problem like the IoT TS process in cloud-fog computing paradigms. Further, the incorporation of diversified number of Swarm Intelligent (SI) optimization algorithms into the real-world problem has facilitated success and exhibited its suitability and ideality in several applications. Many swarm-intelligent bio-inspired optimization schemes are developed for real-time problems. However, during decision making, these algorithms should be enhanced for attaining an improved balance between exploration and exploitation[20]. Further, these algorithms were successfully utilized in the domain of IoT for handling the problems of intrusion detection, sensing applications, feature selection, and so on. Several SI bio-inspired metaheuristic optimization-based TS solutions are proposed over the recent years. But most of them possessed a scope of significant improvement based on makespan, failure rate, execution time and response time during the process of scheduling IoT tasks to available cloud and fog associated VMs in implementation environment. Secretary Bird Optimization Algorithm (SBOA) is one of the recent SI bio-inspired metaheuristic optimization algorithms, which is known for its stability, rate of convergence and searching accuracy. But this classical SBOA algorithm at the first level needs to obtain high accurate solutions with faster convergence. At the second level, global information exchanged between the phases cannot fully adjust the strategy of position updating in the searching space. Finally, it still faces the problem of falling into local optimality when it adopts few simple random strategies of exploitation in the Search Space (SS). Hence, multi-strategy improved CSBOA is indispensable for enhancing the capability of traditional algorithms by utilizing the merits of Feedback Regulation (FR) mechanism, Golden Sinusoidal Guidance (GSG) approach, Co-Operative Camouflage (CC) method and Cosine Similarity (CS)-based update approach for enhancing the rate of exploration and exploitation in a more well-balanced manner. At the same time, RL determines different policies which consolidate the impact of scheduling decisions over time such that it can optimize the intermediate gains. It helps in facilitating optimization which focuses on the long-term objectives that target on maximizing resource utilization, energy consumption and makespan in the fog computing scenario. Motivated by aforementioned merits, an Improved Multi-Strategy Enhanced SBOA and RL (IMSESBOA + RL)-based IoT TS mechanism is proposed for minimizing the delay incurred during mapping tasks to available cloud and fog associated VMs which is highly essential in real-time.

Justification behind selection of SBOA

SBOA is identified as a highly competitive optimization framework, particularly suitable for complex problems such as IoT TS where achieving a fine-grained balance between exploration and exploitation is critical. Many conventional optimization techniques exhibit either excessive exploration, leading to slow convergence or excessive exploitation, which increases the risk of premature convergence. In contrast, SBOA maintains a more adaptive interplay between these two phases. Its intrinsic mechanism enables broad exploration of the solution space during the initial iterations, while progressively shifting focus towards exploiting promising regions as the search advances. This adaptive search behavior results in enhanced global optimization capability and a notably faster convergence rate. Moreover, SBOA demonstrates a strong ability to evade falling into local optima, a common drawback in several existing bio-inspired and swarm-based algorithms. Its dynamic movement patterns and adaptive step-size adjustments contribute to escaping local optimal regions and identifying more diverse and competitive candidate solutions. This capability becomes especially beneficial in IoT TS scenarios, where the optimization landscape is highly dynamic and multimodal.

To further strengthen SBOA's performance, multiple enhancement strategies have been incorporated. Cooperative strategies help individuals share beneficial information, thereby improving population intelligence and reducing redundant search efforts. CS-based updating assists in identifying individuals with correlated search behaviors, guiding the algorithm towards more meaningful directions in the solution space. FR mechanisms stabilize the search dynamics by adjusting parameters in response to intermediate performance. Techniques borrowed from differential evolution introduce controlled perturbations to increase exploration depth, while chaotic maps enhance randomness and help avoid repetitive or stagnated search patterns. These enhancements draw inspiration from powerful optimization frameworks such as PSO, ABC, and the Harris Hawk Algorithm, enabling a hybridized version of SBOA capable of delivering superior optimization quality. Collectively, the integration of these strategies not only enriches solution diversity but also offers improved accuracy, robustness and reliability of optimization outcomes. Consequently, the improved SBOA becomes highly suitable for addressing the complex, time-sensitive, and resource-dependent requirements of IoT TS and similar large-scale optimization problems.

Justification behind adaptability of proposed approach under batch and dynamic workloads

The proposed methodology facilitated adaptive TS under dynamic batch workloads by utilizing the benefits of Enhanced SBOA intelligent algorithm (ESBOA) and iterative learning (RL). It facilitates real time monitoring for adjusting task allocation dynamically depending on the requirements of user QoS, resource availability and workload changes. The use of RL helps in improving the IoT TS process by using execution time and deadline for determining task priority and achieving optimal resource allocation through continuous feedback. This use of RL facilitates the suitable mapping of tasks to fog computing resources in constantly changing environment. It adopted intelligent resource mapping during the implemented system assigns tasks by considering the types of resource pertaining to memory, CPU and their capabilities. It specifically minimized task rejection rates and SLA violations by providing high priority to latency sensitive tasks. It included a dynamic load balancing mechanism during which adaptive scheduling is achieved by distributing the workloads over the available fog computing nodes depending on the execution and current capabilities. This inclusion of dynamic load balancing mechanism prevents bottlenecks and further optimizes resource utilization rate. It performed batch processing under dynamic workloads such that the system is capable for potentially utilizing the resources by processing multiple numbers of tasks together. This batch processing minimized the SS when the IoT tasks to be scheduled have similar dependencies or requirements. This batch processing also enhances matching of resource capacity.

Reasons for selecting ESBOA and integrating RL for boosting TS process

The ESBOA is used for boosting the process of IoT TS due to its capability of fast convergence, stability and maximized diversity of solutions in SS. In specific, the incorporation of multiple strategies into the classical SBOA played an indispensable role in attaining better convergence speed and superior solution quality. It is capable of determining better solutions compared to the baseline optimization algorithms, and it obtains optimal or near-optimal solutions more quickly. Inclusion of multiple strategies helps in establishing balanced exploration and exploitation which searches for new solutions, while refining current solutions in the SS. It possesses a high degree of versatility and adaptability which makes it more ideal for different scenarios of scheduling. Its flexibility also facilitates the option of tailoring with other optimization methodologies for the objective of addressing the challenges associated with scheduling process. Further RL is integrated with ESBOA for the reasons that include (i) it uses agents to make intelligent decisions depending on the past interactions and experiences with the implementation scenario, (ii) The agents learns from rewards and punishments for the purpose of facilitating optimal task allocation methodology and scheduling tasks over different VMs, (iii) it possesses the capability of handling non-deterministic and dynamic situations under which the resource availability and characteristics of tasks are uncertain, and (iv) It adopted a trial and error learning process which can significantly handle uncertain situations and facilitated reliable decisions of scheduling.

Objectives

The core objectives of this research are listed as follows.

i) To implement a dynamic real-TS scheme for executing the bag-of-tasks applications in the Fog-Cloud environment.
ii) To formulate and implement a permutation-based problem of TS over which multi-strategy improved SBOA and RL are used for facilitating different permutations of tasks in every round of the scheduling process.
iii) To allocate the tasks to each of the virtual machine which possesses adequate amount of resources such that minimized execution time is incurred when they are scheduled in the order identified by the best permutations.
iv) To assess the capability of ESBOA-based TS algorithm with respect to evaluation metrics of makespan, execution time, Fitness function, Mean energy consumptions and Mean cost under different number of IoT tasks entering into theFog-Cloud Computing environment.

Major contributions

Major contributions of proposed IMSESBOA-based IoT TS mechanism is listed as follows.

a) Employed a multi-objective methodology using the balanced exploration and exploitation capabilities of SBOA with multi-strategy benefits help in maximizing resource utilization rate and shortening makespan.
b) A blended Improved SBOA using RL with the potential of reducing the time incurred for data processing and improving QoS in fog-cloud computing.
c) Developed as an optimized scheduling model which explores and processes different scalable numbers of tasks by minimizing latency and energy costs.
d) The simulation experiments confirmed better results based on execution time, makespan, energy consumption with scalable number of IoT tasks in contrast to benchmarked schemes under the evaluations done with real and synthetic workloads.

The following sections are structured as follows. Related work Section gives the survey of existing SI bio-inspired metaheuristic optimization algorithms-based TS solutions proposed in recent years with their pros and cons.

Next Section gives a detailed view of the problem formulated, multi-objective optimization model used for IoT TS, and the background details of CSBOA and the adopted Multi strategy-improved SBOA algorithm used for attaining the objective of IoT TS in cloud-fog computing scenario. Results and discussions Section shows the simulation results and interpretations determined from experiments conducted for evaluating the capability of IMSESBOA-based IoT TS scheme based on makespan, execution time, value of fitness function, energy consumptions and cost with change in the number of incoming IoT tasks. Last Section gives the conclusion with major contributions, limitations and future scope of improvement.

Related work

In this section, existing SI bio-inspired metaheuristic optimization algorithms-based IoT TS solutions proposed in recent years are presented with pros and cons.

Mousavi et al.[21] have proposed a Directed Non-Dominated Sorting Genetic Algorithm (DNSGA-II)-based TS mechanisms for addressing issues of latency partially and resolving the shortcomings of IoT-based cloud computing paradigm in a more dynamic manner. This DNSGA-II-based TS mechanism helped in attaining high quality services which maximize system performance such that energy consumptions incurred to the process of mapping computing devices to IoT tasks are comparatively minimized in the implementation environment. It was proposed as a significant solution to bi-objective optimization problem which concentrated on reducing response time and energy consumptions incurred by the servers during the execution of the IoT tasks. It specifically adopted the merits of the recombination operator into the DNSGA-II algorithm for balancing exploration and exploitation stages involved in resource utilization. This adoption of recombination operator also aided in controlling the process of selecting the agents which facilitated the option of optimal mapping of IoT tasks to fog servers within specified deadline considered for execution. Abohamama et al.[22] have proposed a Permutation-Inspired Modified Genetic Algorithm-based TS (PIMGATS) for preventing delay sensitive applications from being executed in the cloud data centers. This PIMGATS approach facilitated the option of extending cloud resources to network edge such that response time of IoT tasks is reduced comparatively during execution. It was proposed as a semi-dynamic TS strategy which uniquely addressed the problems associated with the bag-of-tasks applications by formulating the problem as a permutation-based optimization solution. It specifically used the modified version of GA for generating different possibilities of arrived tasks which need to be scheduled to the computing resources at each round of implementation. It further assigned the tasks depending on the best permutation to the available VMs for the objective of utilizing the necessitated resources that aids in minimizing the required time of execution. The results of this PIMGATS approach confirmed reduced failure rate, elapsed run time, mean delay, total execution time and makespan compared to the best first, first fit, bee and GA algorithms used for comparison.

Further Hussain and Begh[23] have proposed a Hybrid Flamingo Search with a Genetic Algorithm-based TS (HFGATS) technique for facilitating the support that aids in maximizing the degree of QoS and simultaneously minimizing data delay in cloud-fog scenario. This HFGATS technique achieved TS by maintaining the requirements of QoS such that high service latency introduced by occurrence of bursty data traffic never hurdles the process of dynamic TS process in the cloud-fog environment. It boosted the QoS level by concentrating on the process of minimizing the deadline violation cost, communication cost and computational costs such that unnecessary delay is completely prevented in the implementation environment. The statistical results of this HFGATS technique conducted using the Friedman Rank Test confirmed its efficiency in guaranteeing reduced throughput and response time during the execution of IoT tasks. The experiments of this HFGATS technique, conducted using different scalable number of tasks and sizes, confirmed its predominance in attaining the essential level of QoS better than compared RR, Min-V, Min-CCV, PSO, and GA-based TS approaches used for comparison. Agarwal et al.[24] have proposed a GA-based optimized IoT TS mechanism (GAOITSA) which concentrates on minimizing the amount of energy utilized and makespan in cloud-fog computing environment. This approach facilitated energy aware scheduling for the objective of executing the tasks over the processors such that better schedules of tasks are determined in fog-cloud computing in a dynamic manner. It specifically integrated the significance of GA and energy conscious scheduling strategy for preventing computationally expensive application tasks from starving for a considerably longer time compared to conventional TS schemes. It used GA to generate three primary solutions which are explored and exploited well during the process of handling the issues that are associated with energy utilization rates. The results confirmed improved results based on makespan and energy consumptions compared to RR, ACO, GSA and PSO-based TS strategies contributed for executing IoT tasks in fog-cloud environment.

Furthermore Saif et al.[25] have propounded a Multi-Objective Grey Wolf Optimizer Algorithm-based IoT TS Scheme (MOGWATSS) for cloud-fog computing scenario as offloading tasks to fog nodes decreases transmission delay to the required level. This MOGWATSS approach focused on the objectives of energy consumption, delay and QoS for the purpose of distributing the tasks using the fog broker. It was implemented as one of the significant approaches which helped in increasing energy consumption degree, while migrating tasks to the cloud with increased transmission delay. The simulation results confirmed minimized energy consumption and delay independently to the number of tasks entering cloud-fog computing environment. Khiat et al.[26] have proposed another GA-based TS Algorithm (GATSA) for establishing an optimal balance amid the total response time and total consumed energy in fog-cloud-based environment. This GATSA approach is capable of minimizing the latency with reduced energy consumptions by mapping tasks produced from IoT devices to existing fog servers in implementation scenario. It was proposed as one of the predominant scheduling strategies which helped in mapping the resources in a more reasonable time. The investigations of this GATSA scheme conducted using eight datasets with different scalable sizes guaranteed mean improvement in terms of the normalized function used for evaluating process of TS.

In addition, Attiya et al.[27] have proposed a TS Mechanism using merits of Hunger Game Search and Marine Predator Algorithm (HGSMPA) for addressing the limitation associated with the network bandwidth that helped in executing the latency-sensitive applications in fog-cloud environment. HGSMPA dynamically and mutually balanced the benefits of MPA and HGS for establishing the objective of executing the IoT tasks without any delay in cloud-fog computing scenario. It was one among the few methods which not only concentrated on mapping tasks to the resources but also focused on satisfying multiple number of constraints that played a vital role during the determination of schedules. The results of the HGSMPA scheme confirmed lesser makespan and energy consumptions with respect to scalable number of tasks derived from the real workloads during the scheduling process. With respect to synthetic workloads, this HGSMPA approach also confirmed the same pattern based on energy consumption and makespan compared to baseline schemes taken for assessment. Further, Salehnia et al.[28] have propounded a Multi-Objective Moth-Flame Optimization Algorithm-based IoT task Request Scheduling Method (MOMFOA) for accelerating the rate of storing and processing the big data aggregated in the fog-cloud environment. This MOMFOA approach contributed towards minimized energy consumption, throughput rates and tasks' request completion time for the objective of enhancing the IoT services quality which is highly essential in fog-cloud computing system. It was proposed with a strategy which emphasizes that diminishes in the percentage of carbon emissions proportionally decreases rate of energy consumption in implementation scenario. The results of this MOMFOA approach were identified to increase the rate of system performance with minimized energy consumption, carbon emissions and delay while processing of IoT tasks in fog-cloud computing scenario.

Ali and Sridevi[29] have proposed an IoT TS algorithm using mobility for enhancing the degree of processing time when the process of real-time tasks is imposed for execution in the fog-cloud computing environment. This MSTSM-based TS model was proposed with the requirements of security and resource requirements which need to be handled with the task time constraints. It first focused on the issue of mobility which aided in assigning the tasks generated by IoT devices by including factors of load, bandwidth and distance into account. It specifically used fuzzy logic to handle uncertainty such that TS can be optimized by introducing ideal distribution of tasks over the fog-cloud scheduling by adopting the requirements of tasks security. It exploited the deadlines and demands of the tasks for selecting the proper and suitable processing unit for selecting tasks in the fog computing layer. The results of MSTSM-based TS model confirmed better processing time, success ration, turnaround time and makespan.

Salimi et al.[30] have proposed a greedy randomized adaptive search algorithm (GRASA) -based IoT TS algorithm is proposed for effectively utilizing the VMs of the cloud and fog for satisfying user demands. This GRASA approach was formulated as a reliable strategy which addressed the problem of heterogeneity which is realized in terms of energy consumption, communication delay and processing power. It was proposed with the capability of scheduling the tasks by due consideration of overall energy consumption and individual task deadlines. It specifically used a randomized greedy approach for attaining optimal allocation of tasks to suitable VMs of fog-cloud scenario. It utilized only a restricted number of tuning factors with implementation ease and simplicity for the objective of attaining optimized tasks allocation. The results confirmed minimized makespan, reduced energy consumption and shortened mean response time with due satisfaction of deadlines in the implementation domain.

Vijayalakshmi and Saravanan[31] have proposed a RL-based multi-objective energy efficient TS approach for satisfying the requirements of the users interacting in the IoT systems. This TS algorithm used ID3 algorithm for classifying the tasks depending on the process requirements, QoS and priority. Then it selected suitable fog nodes after the classification process with respect to the factors associated with the tasks under execution using proximity of IoT nodes, processing capability and energy consumption of nodes in the fog computing scenario. It further used first fit algorithm for offloading the unhandled tasks from the fog nodes to the cloud data centers depending on the location of fog node, requirements of QoS and available resources. Finally, reinforcement algorithm is incorporated for scheduling the tasks for execution once the required cloud data centers or fog nodes are identified in the network. The results of this RL-based multi-objective energy efficient TS approach confirmed better resource management, energy efficiency, task completion rate and accuracy rate compared to the benchmarked Round Robin, FCFS and SJF methods.

Wang et al.[32] have proposed hierarchical adaptive federated RL approach for achieving reliable scheduling and resource allocation in fog computing environment. It was proposed with capability of handling stochastic and dynamic properties of IoT applications. This RL approach was implemented to limit the computational burden over fog computing systems such that the requirements of the users on demand are satisfied with better optimality. It was proposed as an adaptive and efficient optimizing strategy which optimizes the response time required for the execution of heterogeneous IoT application such that load is balanced in the fog computing systems. It was implemented as a real time scheduler with the support of the FogBus2 function which helped in attaining a service framework to create fog-cloud blended serverless computing scenario. The results of this RL mechanism confirmed minimized weighted cost, response time and load balancing compared to the baseline approaches.

In addition, Choppara and Mangalampalli[33] have proposed an Integrated deep RL and fuzzy logic-based TS method for achieving efficient processing and managing of data in the fog cloud computing scenario. It was propounded with efficiency of fulfilling the needs of IoT nodes in terms of real time processing, reduced makespan, and optimized bandwidth utilization. It specifically used a Takagi-Sugeno fuzzy inference system. Continuous interaction with the environment is to be facilitated such that tasks can be prioritized while scheduling tasks in fog computing scenario. It handled the continuously changing requirements of the IoT tasks through the real-time change of the rules associated with the scheduling process. The results of RL scheme confirmed better performance based on fault tolerance, cost, energy consumption and makespan.

Extract of the literature

Following shortcomings are identified from the survey on existing metaheuristic optimization algorithms-based IoT TS approaches.

i) It faces the challenge during the process of optimizing resource utilization, since it poses the problem of managing the complexity involved during the management of distributed resources.
ii) It encounters the challenge during the process of guaranteeing real-time performance for time-sensitive IoT applications, since it struggles in establishing a balance between the latency with respect to resource consumptions.
iii) It poses the difficulty of managing heterogeneous resources with varying capabilities such that well balance of energy consumption and cost is attained in the cloud-fog computing scenario under implementation.
iv) The IoT TS methods that utilize a standalone metaheuristic optimization algorithm (Scheduling algorithms that used PSO, GA and MFOA) suffered from issues of premature convergence, poor solution diversity and increased chance of falling into local point of optimality during load balancing and resource utilization..
v) The existing hybrid metaheuristic optimization algorithms-based TS approaches (Scheduling approaches that used HGS-MPA and LF-MFOA faced the challenge of integrating conflicting factors that contribute towards efficient and effective IoT TS process.
vi) The existing IoT TS approaches generally are semi-dynamic and hence the delay incurred during the process of resource allocation gets peaked when large amount of data is generated from the input determining environment.
vii) Majority of the recently developed metaheuristic optimization algorithm based IoT TS approaches still possessed a scope of enhancing the degree of throughput with minimized delay, energy consumptions and failure rate.

An improved Multi-Strategy enhanced SBOA (IMSESBOA)-based IoT TS mechanism

In this section, initially problem formulation of IoT TS Mechanism used for implementing the proposed IMSESBOA approach is presented. Then the detailed view of the classical SBOA and its enhanced version used for attaining IoT TS process in cloud-fog computing scenario is presented in Fig. 2.

Problem statement

This problem of IMSESBOA approach-based IoT TS Mechanism works on the considered three layered architectures of fog, cloud and IoT devices for determining the schedules using task scheduler (that mimics the characteristics of ESBOA) that helps in allocating the tasks to respective computing nodes. This TS used the benefits of ESBOA for exploring and exploiting the properties of the tasks produced by IoT devices with consideration with the potential ties of the resources existing in the fog-cloud computing model. This approach was offered as a reliable attempt in satisfying the processing and storage requirements of the requests associated with the IoT devices which need to be sent to the higher layers of cloud and fog. It facilitates the option of storing and processing time-sensitive tasks which are much closer to the devices for reducing delay by adopting fog computing. It also used the methodology of sending computational-intensive tasks to the cloud servers since the cloud servers facilitate better storage and computing capabilities compared to the fog nodes.

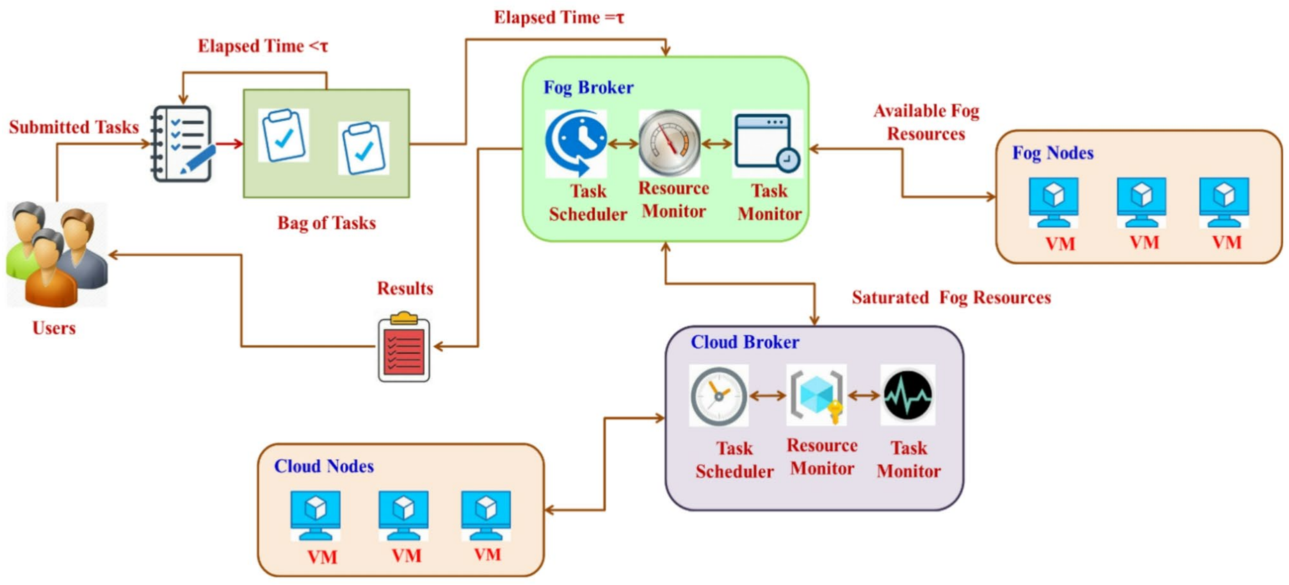


Fig. 2. Overall view of the proposed IMSESBOA-based IoT TS Mechanism

Problem formulation

In fog-cloud computing environment, let ‘ n’ denote the number of real-time delay sensitive tasks (the tasks are independent of each other in cloud-fog environment) with m number of heterogeneous virtualized cloud node and z as the number of heterogeneous virtualized fog nodes. The problem of dynamic TS concentrates on the process of allocating each of the submitted IoT devices generated tasks to appropriate VMs in the fog-cloud environment with the constraints imposed during the implementation process. If each of the VMs are allocated with one or more tasks during execution, then the tasks list allocated to VMs is represented using $[\mathrm{T}_{\mathrm{sk}(1),}\ \mathrm{T}_{\mathrm{sk}(2)}, \ldots\ \mathrm{T}_{\mathrm{sk}(z)}, \ldots .., \mathrm{T}_{\mathrm{sk}(n)}]$. At this juncture, the proposed IMSESBOA approach helps in determining the suitable schedule through which the IoT tasks can be executed within the expected time of task completion ($\mathrm{FT}_{\mathrm{Exp}}$) determined based on Eq. (1)

$$\mathrm{FT}_{\mathrm{Exp(i)}} = \mathrm{AT}_{\mathrm{Br(i)}} + \mathrm{RT}_{\mathrm{TK(i)}} + \mathrm{WT}_{\mathrm{ij}} + \mathrm{ET}_{\mathrm{Exp(i)}} \tag{1}$$

Where, $\mathrm{FT}_{\mathrm{Exp(i)}}$ and $\mathrm{AT}_{\mathrm{Br(i)}}$ represents the completion time expected for each of the i^{th}task to be executed on j^{th} VM and arrival time of the broker (can be cloud or fog broker) during the process of scheduling the i^{th}task. Further $\mathrm{RT}_{\mathrm{TK(i)}}$ and $\mathrm{WT}_{\mathrm{ij}}$ represents the i^{th}task remaining time of execution and waiting time of i^{th}task to wait in the queue before it is scheduled over the cloud VMs or fog VMs in cloud-fog environment. Moreover, $\mathrm{ET}_{\mathrm{Exp}}$indicates the execution time expected for each of i^{th}task to be executed in j^{th} VM. Then, i^{th} task remaining time of execution ($\mathrm{RT}_{\mathrm{TK(i)}}$) is determined based on Eq. (2)

$$\mathrm{RT}_{\mathrm{TK(i)}} = \mathrm{AT}_{\mathrm{N(i)}} - \mathrm{AT}_{\mathrm{Br(i)}} \tag{2}$$

Where $\mathrm{AT}_{\mathrm{N(i)}}$ is the i^{th}task arrival time (node arrival time) at selected j^{th} VM. Further the $\mathrm{ET}_{\mathrm{Exp}}$ representing the execution time expected for each of the i^{th}task to be executed in the j^{th} VM is computed based on Eq. (3)

$$\mathrm{ET}_{\mathrm{Exp}} = \frac{\mathrm{FS}_{\mathrm{IN\ (i)}} + \mathrm{FS}_{\mathrm{OUT(i)}}}{\mathrm{VM}_{\mathrm{BW(j)}}} + \frac{L_{tsk(I)}}{CE_{CP(VM(j))} * N_{CE(MIPS(j))}} \tag{3}$$

Where, $\mathrm{FS}_{\mathrm{IN\ (i)}}$and $\mathrm{FS}_{\mathrm{OUT(i)}}$ denotes input and output file size associated with i^{th}task executed in the selected j^{th} VM. Then $\mathrm{VM}_{\mathrm{BW(j)}}$and $L_{tsk(I)}$ indicates the bandwidth related to j^{th} VM and length of i^{th}task (in million instructions). In addition, $CE_{CP(VM(j))}$ and $N_{CE(MIPS(j))}$ represents the computing elements' processing capability associated with each of the j^{th} VM (in million instructions per second (MIPS)) and number of processing elements associated with the same j^{th} VM.

In this context, if $\mathrm{FT}_{\mathrm{Exp(ij)}} \leq \mathrm{DL}_{\mathrm{tsk(i)}}$ is satisfied, then i^{th}task is executed in the selected j^{th} VM. Else the i^{th}task is failed, and it is not executed on the selected j^{th} VM. Where $\mathrm{DL}_{\mathrm{tsk(i)}}$is the deadline for executing each of the tasks in the fog nodes. The rate of failure ($\mathrm{FR}_{\mathrm{TSK}}$) with respect to the i^{th}task is determined based on Eq. (4)

$$\mathrm{FR}_{\mathrm{TSK}} = \frac{\mathrm{N}_{\mathrm{FL(TSK)}}}{\mathrm{N}_{TSK}} \tag{4}$$

Where $\mathrm{N}_{\mathrm{FL(TSK)}}$ and N_{TSK} represents number of tasks failed and total number of tasks which need to be executed in available VMs (cloud and fog VMs). When $RT^{n}_{TSK}$is the final task allocated to j^{th} VM, then time essential for completing the execution of tasks ($\mathrm{ET}_{\mathrm{TSK(j)}}$) allocated to j^{th} VM is determined based on Eq. (5)

$$\mathrm{ET}_{\mathrm{TSK(j)}} = \mathrm{FT}_{\mathrm{Exp(ij)}} \tag{5}$$

At this juncture, Makespan is defined as the total time necessitated by the system for completing the execution of all the tasks in the fog-cloud environment, then it is computed based on Eq. (6)

$$\mathrm{Makespan} = \max_{1 \leq \mathrm{j} \leq \mathrm{m}} \mathrm{ET}_{\mathrm{TSK(j)}} \tag{6}$$

Where m denotes the total number of VMs considered for executing the complete set of tasks in the cloud-fog environment. Further $\mathrm{MinMakespan}$ is defined as the minimized time required for completing all the tasks (which is the lower threshold of the makespan)

$$\mathrm{MinMakespan} = \begin{cases} \dfrac{\sum_{i=1}^{n} \mathrm{FS}_{\mathrm{IN\ (i)}} + \mathrm{FS}_{\mathrm{OUT(i)}}}{\sum_{j=1}^{m} \mathrm{VM}_{\mathrm{BW(j)}}} + \dfrac{\sum_{i=1}^{n} L_{tsk(I)}}{\sum_{j=1}^{m} CE_{CP(VM(j))} * N_{CE(MIPS(j))}} & \text{If k} = 1 \\ \dfrac{\sum_{i=1}^{n} \mathrm{FS}_{\mathrm{IN\ (i)}} + \mathrm{FS}_{\mathrm{OUT(i)}}}{\sum_{j=1}^{m} \mathrm{VM}_{\mathrm{BW(j)}}} + \dfrac{\sum_{i=1}^{n} L_{tsk(I)}}{\sum_{j=1}^{m} CE_{CP(VM(j))} * N_{CE(MIPS(j))}} + \dfrac{(k*\delta + (k-1)*\delta)}{2} & \text{If k} > 1 \end{cases} \tag{7}$$

Where k which lies between 1 and ∞ indicates current TS round with n and m as total number of tasks coming into first round of scheduling until the final k^{th}scheduling round and number of VMs in cloud-fog computing environment. The value of $k = 1$ denotes the first scheduling round which starts at time 0 for all the complete set of tasks whose arrival time of the broker is equal to 0. In specific, the term $\frac{(k*\delta + (k-1)*\delta)}{2}$ helps in including the arrival time of different brokers with respect to the execution of each task in every round of scheduling. In other words, it indicates the mean arrival time of broker with respect to the tasks considered in the present round of scheduling. Hence the broker arrival times of tasks are represented using $\mathrm{AT}_{\mathrm{Br(i)}} \leq (k-1)*\delta$, such that the scheduling times is determined based $\mathrm{Sh_{Time}} = (\mathrm{k}-1)^{*}\delta$, respectively. This term k also represents the condition when the number of rounds is greater than 1.

In this cloud-fog computing environment of implementation, the monetary cost used for attaining the required objective of IoT TS completely depends on cost with respect to bandwidth, memory and processing. In this proposed IMSESBOA-based IoT TS Mechanism, a model of pricing which is similar to the pricing model of Alibaba cloud resource is considered for determining the cost of each resource as a separate entity. It is noted that the cost of bandwidth consumption is completely different from the computing power utilization or memory utilization. Then the cost of executing the $\mathrm{i^{th}}$task ($\mathrm{C_{E(TSK(ij))}}$) to be executed in the j^{th} VM is calculated using $\mathrm{C_{ij}^{CPU}}$, $\mathrm{C_{ij}^{RAM}}$ and $\mathrm{C_{ij}^{BW}}$as presented in Eq. (8)

$$\mathrm{C_{E(TSK(ij))}} = \mathrm{C_{ij}^{CPU}} + \mathrm{C_{ij}^{RAM}} + \mathrm{C_{ij}^{BW}} \tag{8}$$

Then the cost of CPU cost ($\mathrm{C_{ij}^{CPU}}$) is determined based on processing cost per unit cost ($\mathrm{PC_j}$) and expected time of execution ($\mathrm{ET_{Exp}}$) as specified in Eq. (9)

$$\mathrm{C_{ij}^{CPU}} = \mathrm{PC_j}^{*}\mathrm{ET_{Exp}} \tag{9}$$

Further the cost of memory unit ($\mathrm{C_{ij}^{RAM}}$) is determined based on memory requirements ($\mathrm{MR_{T(i)}}$) of the $\mathrm{i^{th}}$ task and memory cost per storage unit ($\mathrm{MC_{SU(j)}}$) associated with the j^{th} VM as mentioned in Eq. (10)

$$\mathrm{C_{ij}^{RAM}} = \mathrm{MC_{SU(j)}}^{*}\mathrm{MR_{T(i)}} \tag{10}$$

In addition, the bandwidth cost ($\mathrm{C_{ij}^{BW}}$) is determined based on bandwidth requirements ($\mathrm{BWR_{(i)}}$) of the $\mathrm{i^{th}}$ task and bandwidth cost per data unit ($\mathrm{BC_{DA(VM(j))}}$) associated with the j^{th} VM as mentioned in Eq. (11)

$$\mathrm{C_{ij}^{BW}} = \mathrm{BC_{DA(VM(j))}}^{*}\mathrm{BWR_{(i)}} \tag{11}$$

Then the requirements of bandwidth $\mathrm{BWR_{(i)}}$ is calculated based on Eq. (12)

$$\mathrm{BWR_{(i)}} = \mathrm{FS_{IN\ (i)}} + \mathrm{FS_{OUT(i)}} \tag{12}$$

As mentioned earlier, $\mathrm{FS_{IN\ (i)}}$and $\mathrm{FS_{OUT(i)}}$signifies the input and output file size associated with the $\mathrm{i^{th}}$task executed in the selected $\mathrm{j^{th}}$ VM. Then, total cost incurred ($\mathrm{EC_{Total}}$) for implementing the complete set of tasks in the cloud-fog computing scenario is computed based on Eq. (13)

$$\mathrm{EC_{Total}} = \sum_{\mathrm{i=1}}^{\mathrm{n}} \sum_{\mathrm{j=1}}^{\mathrm{m}} \mathrm{B_V}^{*}\mathrm{C_{E(TSK(ij))}} \tag{13}$$

Where, $\mathrm{B_V}$ is the binary value which is set to 0 ($\mathrm{i^{th}}$task is not allocated to the selected $\mathrm{j^{th}}$ VM) or 1 ($\mathrm{i^{th}}$task is allocated to the selected $\mathrm{j^{th}}$ VM).

If $\mathrm{Min\ Cost_j}$is the least cost required for executing tasks in the given cloud-fog computing model with the available cost information associated with diverse VMs, then the tasks executed on the least cost node is determined based on Eq. (14) and Eq. (15).

$$\mathrm{Min\ Cost_j} = \min_{1\leq \mathrm{j} \leq \mathrm{m}} \mathrm{C_{E(TSK(ij))}} \tag{14}$$

$$\mathrm{Min\ EX_{Cost}} = \sum {}_{\mathrm{i=1}}^{\mathrm{n}} \mathrm{Min\ Cost_j} \tag{15}$$

Based on the above facts, the dynamic TS model used for implementing the proposed IMSESBOA-based IoT TS Mechanism is formulated as the problem of optimization as specified in Eq. (16)

$$\mathrm{Obj_{Fn}} = \mathrm{Maximize}\left(\frac{\mathrm{MinMakespan}}{\mathrm{Makespan}}, \frac{\mathrm{Min\ EX_{Cost}}}{\mathrm{EC_{Total}}}\right) \tag{16}$$

In most scenarios of optimization, the value of $\mathrm{MinMakespan}$ may be equal to $\mathrm{Makespan}$, and the value of $\mathrm{Min\ EX_{Cost}}$ may also be equal to the value of $\mathrm{EC_{Total}}$. Hence the upper threshold of every term used in Eq. (16) may be equal to 1.

In specific, Fig. 3 demonstrates process of TS in cloud-fog computing environment.

Secretary bird optimization algorithm (SBOA)

Classical SBOA (CSBOA) was proposed by Fu et al.[34] inspired by the foraging behavior of secretary birds which are termed as large terrestrial raptors that are present in semi-desert and savannas regions. It was specifically proposed based on their inspiration during their process of searching prey and evading search from its predators. This CSBOA algorithm is identified to be more suitable for addressing the problems of complex optimization which are common in solving in engineering problems such as the process of IoT TS problem which possesses a NP-hard complexity. The mathematical modeling of the utilized CSBOA algorithm is presented as follows.

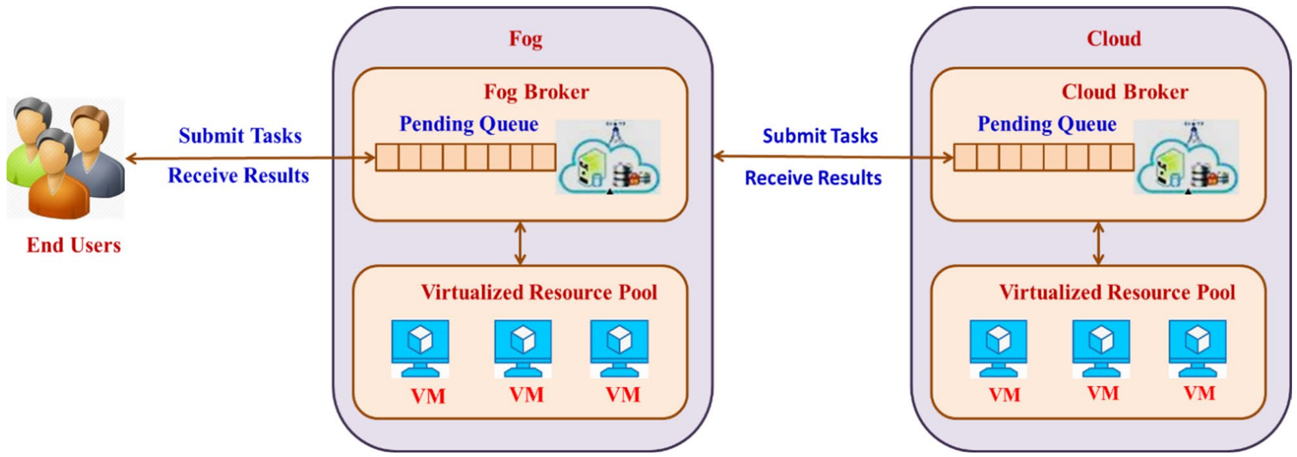


Fig. 3. Process of TS in cloud-fog computing environment

Phase of initialization

The initial solution considered for starting the process of searching with the objective of mapping incoming IoT tasks to available resources in fog-cloud environment is identified. This phase is accountable for determining n number of initial candidate solutions' population through the use of 'n' search agents (secretary birds). These search agents evaluate the n number of initial candidate solutions $C_{SB} = \left[C_{SB(1)}, C_{SB(2)}, \ldots \ldots, C_{SB(n)},\right]$ using d number of variables or dimensions considered for solving the problem of optimization. In this context, each candidate solution $C_{SB(i)}$ corresponds to the solution identified by the i^{th}search agent (secretary bird) as presented in Eq. (17)

$$C_{SB(i)} = L_{BD} + rand_{[0,1]} * (U_{BD} - L_{BD}) \tag{17}$$

Where, U_{BD} and L_{BD} represent upper and lower threshold values of 'd' number of variables or dimensions considered for solving the problem of optimization. Then, $rand_{[0,1]} \in [0, 1]$ is a random number. At this juncture, the complete candidate solution matrix determined by 'n' search agents (secretary birds) with respect to 'd' variables is presented in Eq. (18)

$$C_{SB(Matrix)} = \begin{bmatrix} sb_1^1 & sb_1^2 & \cdots & sb_1^d \\ sb_2^1 & sb_2^2 & \cdots & sd_2^d \\ \vdots & \vdots & \cdots & \vdots \\ sd_n^1 & sd_n^2 & \cdots & sd_n^m \end{bmatrix} \text{where } 1 \leq i \leq n \text{ and} 1 \leq j \leq d \tag{18}$$

Then the fitness value of each candidate solution in SS during each iteration is determined based on Eq. (19)

$$F(SC_{ij}) = \begin{bmatrix} F(sb_1^1 & sb_1^2 & \cdots & sb_1^d) \\ F(sb_2^1 & sb_2^2 & \cdots & sd_2^d) \\ \vdots & \vdots & \cdots & \vdots \\ F(sd_n^1 & sd_n^2 & \cdots & sd_n^m) \end{bmatrix} \text{where } 1 \leq i \leq \text{ and } \leq j \leq d \tag{19}$$

Phase of exploration

This exploration phase of CSBOA mimics the hunting strategy adopted by the secretary birds during the process of searching, attacking and consuming the prey (snakes). The mathematical model with respect to the three steps of exploration such as searching, attacking and consuming the prey is presented as follows.

Prey searching

In this first step of exploration, sufficient amount of information associated with the process of searching is acquired. Then depending on this information acquired from the SS, the search agents explore new regions by considering the positions of the other search agents. To achieve this process of exploring the new regions, operations associated with differential mutation is used for enhancing the solution diversity of the algorithm. Thus each of the i^{th}search agent uses Eq. (20) and Eq. (21) for updating its position when present iteration count is less than one-third of maximized number of iterations ($Iter_{Curr} < \frac{1}{3} Iter_{Max}$).

$$C_{SB}{}_{(i)}^{PS-New(t)} = C_{SB(i)}(t) + \left(C_{SB(r1)}(t) - C_{SB(r2)}(t)\right) \times R_{V(1)} \tag{20}$$

$$\mathrm{C_{SB}}_{(i)}^{t+1} = \begin{cases} \mathrm{C_{SB}}_{(i)}^{\mathrm{PS-New(t)}} & \text{If } \mathrm{F_i^{New}} < \mathrm{F_i} \\ \mathrm{C_{SB}}_{(i)}^{t} & \text{Otherwise} \end{cases} \tag{21}$$

where $\mathrm{C_{SB(r1)}}\,(t)$ and $\mathrm{C_{SB(r2)}}\,(t)$ are two individual solutions randomly chosen from present population with $\mathrm{R_{V(1)}}$ as the random vector that included 1 × M elements chosen from the range [0, 1].

Prey consumption

In this second step of exploration, the search agents perform the global optimization process in which they search for better candidate solutions in SS. This behavior of the search agent (secretary birds) corresponds to the Brownian motion of encircling the candidate solution depending on the random number whose dimensions lies between $[1, d]$ under the condition $\frac{1}{3}\mathrm{Iter_{Max}} \le \mathrm{Iter_{Curr}} \le \frac{2}{3}\mathrm{Iter_{Max}}$ as specified in Eq. (22)

$$\mathrm{BR_M} = \mathrm{Rand}[1, d] \tag{22}$$

Where, $\mathrm{Rand}[1, d]$ is the vector generated randomly by satisfying the properties of the classical normal distribution whose mean value and standard deviation are 0 and 1 respectively.

$$\mathrm{C_{SB}}_{(i)}^{\mathrm{PC-New(t)}} = \mathrm{C_{SB(i)}}\,(t) + \left(\mathrm{C_{SB(r1)}}\,(t) - \mathrm{C_{SB(r2)}}\,(t)\right) \times \mathrm{R_{V(1)}} \tag{23}$$

$$\mathrm{C_{SB}}_{(i)}^{t+1} = \begin{cases} \mathrm{C_{SB}}_{(i)}^{\mathrm{PC-New(t)}} & \text{If } \mathrm{F_i^{New}} < \mathrm{F_i} \\ \mathrm{C_{SB}}_{(i)}^{t} & \text{Otherwise} \end{cases} \tag{24}$$

As mentioned in Eq. (24a), when fitness value of current solution $\mathrm{F_i^{New}}$ is less than the fitness of previous solution ($\mathrm{F_i}$), then the position of present solution in the prey consumption phase ($\mathrm{C_{SB}}_{(i)}^{\mathrm{PC-New(t)}}$) is updated based on Eq. (23). Else the previous solution is considered as the current solution as specified in Eq. (24b).

Prey attacking

In this prey attacking phase is executed when the value of $\mathrm{Iter_{Curr}} \ge \frac{2}{3}\mathrm{Iter_{Max}}$ as specified in Eq. (25) depends on the method of levy flight.

$$\mathrm{C_{SB}}_{(i)}^{\mathrm{PA-New\ (t)}} = \mathrm{C_{SB(i)}}\,(t) + \left(\mathrm{C_{SB(r1)}}\,(t) - \mathrm{C_{SB(r2)}}\,(t)\right) \times \mathrm{R_{V(1)}} \tag{25}$$

$$\begin{cases} \mathrm{C_{SB}}_{(i)}^{t+1} = \mathrm{C_{SB}}_{(i)}^{\mathrm{PA-New(t)}} & \text{If } \mathrm{F_i^{New}} < \mathrm{F_i} \\ \mathrm{C_{SB}}_{(i)}^{t+1} = \mathrm{C_{SB}}_{(i)}^{t} & \text{Otherwise} \end{cases} \tag{26}$$

As specified in Eq. (26a), when fitness of current solution $\mathrm{F_i^{New}}$is less than fitness of previous solution $\mathrm{F_i}$, then position of current solution in prey attacking phase ($\mathrm{C_{SB}}_{(i)}^{\mathrm{PC-New(t)}}$) is updated based on Eq. (25). Else the previous solution is considered as the current solution as specified in Eq. (26b).

Limitations of the classical CSBOA algorithm

The classical CSBOA algorithm, despite several merits, still possesses several limitations with respect to the process of determining high accuracy solutions and faster convergence rate. In specifically, the four main limitations of the classical CSBOA algorithm include (i) improper use of global information during the process of adjusting the strategy of position updating, (ii) it ignores the prey performance during the employment of attack mode associated with the hunting phase of SBOA, (iii) simple random methods adopted during the process of preventing premature convergence, and (iv) imbalanced exploration and exploitation while attaining the objective. These components are essential for maintaining an optimal balance between exploration and exploitation, preventing early convergence to local optima, and improving the overall exploration capability by promoting superior solution diversity. Hence, IMSESBOA is adopted using the merits of FR mechanism, GSG strategy, CC strategy and CS-based update strategy for handling the issues of CSBOA and at the same time, improving the process of IoT TS in the cloud-fog computing environment.

Improved Multi-Strategy enhanced SBOA

In this section, the detailed view of the utilized Improved Multi-Strategy Enhanced SBOA is presented with the merits introduced by them during the process of IoT TS in the cloud-fog computing system.

Enhancement 1: strategy of feedback regulation (FR)

This strategy of FR mainly focuses on the process of determining the possible state of IoT tasks allocation to the available cloud and fog VMs in the immediate state depending on the condition of the current state. This regulation mechanism receives feedback for decision making with respect to the allocation of tasks to the existing resources as specified in Eq. (27)

$$\mathrm{C_{SB(i)}}\,(t) = \mathrm{C_{SB(i)}}\,(t-1) + \mathrm{C_{SB(Best)}}\,(t) - \mathrm{C_{SB(Best)}}\,(t-1) \tag{27}$$

Where $\mathrm{C_{SB(Best)}}\,(\mathrm{t})$ and $\mathrm{C_{SB(Best)}}\,(\mathrm{t}-1)$ represents the best candidate solution determined in the current and previous iteration during the process of FR. Candidate solution is determined in the current and previous iteration during the process of exploration.

This strategy of FR assists in determining the optimal allocation of IoT tasks into the suitable VMs of the fog-cloud environment in each iteration such that outcome of one iteration is used for enhancing the process of improving the optimal resource allocation in the subsequent phases. It explores and exploits the SS in a more reliable manner such that resource management is better facilitated during the scheduling process. In addition, the regulation of feedback helps in achieving more optimized solutions under the enforcement of deadline constraints employed during the optimization process.

Enhancement 2: strategy of golden sinusoidal guidance (GSG)

This strategy of GSG is mainly for utilizing the merits of the location information associated with the objective (TS process) such that the success rate is significantly improved in the implementation scenario (cloud-fog computing system). This enhancement mimics the process of how the secretary bird attacks the prey after locking them. This strategy is also useful in improving the potential of developing local regions which need to be significantly exploited in the SS for determining different feasible solutions (different feasible solutions which represent the mapping of tasks to the available VMs (fog or cloud system)). With respect to each of the search agents in the population, the candidate solutions are updated using GSG strategy as specified in Eq. (28)

$$\mathrm{C_{SB(i)}}(t+1)=\mathrm{C_{SB(i)}}\,(t)+\left|\sin\left(r_{1[0,2\pi\,]}\right)\right|+r_{2[0,\pi\,]}\times\ \left|\sin\left(r_{1[0,2\pi\,]}\right)\right|\times\ \left|\delta_{\,1}\times\ \mathrm{C_{SB(Best)}}\,(t)+\delta_{\,2}\times\ \mathrm{C_{SB(Best)}}\,(t)\right| \tag{28}$$

Where $r_{1[0,2\pi\,]}$ and $r_{2[0,\pi\,]}$ represents the first and second random value that lies between 0 to 2π , and 0 to π , respectively. Further $\delta_{\,1}$and $\delta_{\,2}$ indicates the GSG coefficient. In specific, the value of $\delta_{\,1}$and $\delta_{\,2}$ are determined based on Eqs. (29) and (30)

$$\delta_{\,1}=-\pi\ +2\pi\,(1-\tau\,) \tag{29}$$

$$\delta_{\,2}=-\pi\ +2\pi\,\tau \tag{30}$$

Where the value of τ is computed using $\tau\ =\frac{1-\sqrt{5}}{2}$.

In specific, GSG method plays an indispensable role in enhancing the rate of determining the optimal solution by extracting the local information in the SS such that overloaded or underloaded conditions of VMs during the process of resource allocation is prevented under execution of IoT tasks in a more balanced manner over the fog-cloud computing environment. This GSG method wide opened the option of integrating the merits of golden section search which is a numerical method that helps in identifying the minimum of the function. It targeted the process of enhancing the potential of the algorithm such that potential regions of the SS are explored to attain the guidance towards the process of determining an optimal solution.

Enhancement 3: strategy of cooperative camouflage (CC)

This strategy of CC is mainly for exchanging information between the individuals of the population which is highly ignored in the CSBOA algorithm. This phase mimics how the secretary bird feels threatening, and how they disguise themselves with the use of the existing environment. This strategy is implemented by considering different positions of the individuals (say $\mathrm{C}_{\mathrm{SB}(a)}$, $\mathrm{C}_{\mathrm{SB}(b)}$ and $\mathrm{C}_{\mathrm{SB}(c)}$) such that a new candidate solution is generated based on Eq. (31)

$$\mathrm{C_{SB(i)}}\,(t+1)=\mathrm{C}_{\mathrm{SB}(a)}+r_{v[0,1]}\left|\mathrm{C}_{\mathrm{SB}(b)}-\mathrm{C}_{\mathrm{SB}(c)}\right| \tag{31}$$

Where $r_{v[0,1]}$ is the random variable that lies between 0 and 1.

This CC strategy helps the search agents to work together such that they cooperatively balance exploration and exploitation rates so that optimal solutions are obtained at a rapid rate without any delayed convergence. This method is crucial for avoiding premature convergence to local optima, thereby enabling significant reductions in energy consumption and cost during IoT task execution within real-world implementation scenarios. It also facilitated a balanced search such that potential regions are only exploited during the process of IoT tasks migration under the execution scenario.

Enhancement 4: strategy of cosine similarity (CS) updating

The CS-based updating strategy is employed to determine the probability that facilitates escaping from local optima, thereby improving the robustness of the solution search process. This strategy mimics the pattern through which the secretary birds escape from their enemies by moving intelligently to an empty space. This strategy avails a better region of searching over which potential solutions (mapping possibilities of tasks to resources are achieved) in the SS. The CS which is used for assessing the degree of crowding with respect to different candidate solutions as seen from the perspective of the search agents are determined based on two vectors ($\mathrm{V_A}$ and $\mathrm{V_B}$) as specified in Eqs. (32) and (33)

$$\mathrm{V_A}=\mathrm{C_{SB(i)}}\,(\mathrm{t})-\mathrm{C_{SB(Best)}}\,(\mathrm{t}) \tag{32}$$

$$\mathrm{V_B}=\mathrm{C_{SB(j)}}\,(\mathrm{t})-\mathrm{C_{SB(Best)}}\,(\mathrm{t}) \tag{33}$$

Where the solutions considered for similarity are completely different ($i \neq j$). Then the similarity between $\mathrm{V_A}$ and $\mathrm{V_B}$ is calculated based on Eq. (34)

$$\phi_{\ i,j} = \frac{V_A \cdot V_B}{\lceil V_a \rceil \, |V_B|} \tag{34}$$

Once the value of $\mathrm{C_{SB(i)}}$ is compared with the other existing solutions, then the solution with the least CS is considered during this strategy of CS updating such that the direction of updating will be considered as $\mathrm{C_{SB(s)}}$. Hence the new solution $\mathrm{C_{SB(s)}}$ is determined using CS updating through Eq. (35)

$$\mathrm{C_{SB(i)}}\ (\mathrm{t}+1) = \mathrm{C_{SB(Best)}} + \mathrm{R} \times \ \left(\mathrm{C_{SB(s)}} - \mathrm{T_{IR}}{*}\mathrm{C_{SB(i)}}\right) \tag{35}$$

Where $\mathrm{T_{IR}}$ and R is the toggling integer random that lies between 1 and 2, and a vector that includes random elements that lie between 0 and 1.

This strategy of CS Updating is used for determining how the current solution is changed compared to the previous solution in terms of magnitude such that the positions of the current solution need be replaced is identified. It helps in determining the degree to which the current solutions' position is updated in the SS.

In addition, Fig. 4 presents the flowchart of the adopted IMSESBOA used for IoT TS in the cloud-fog computing environment.

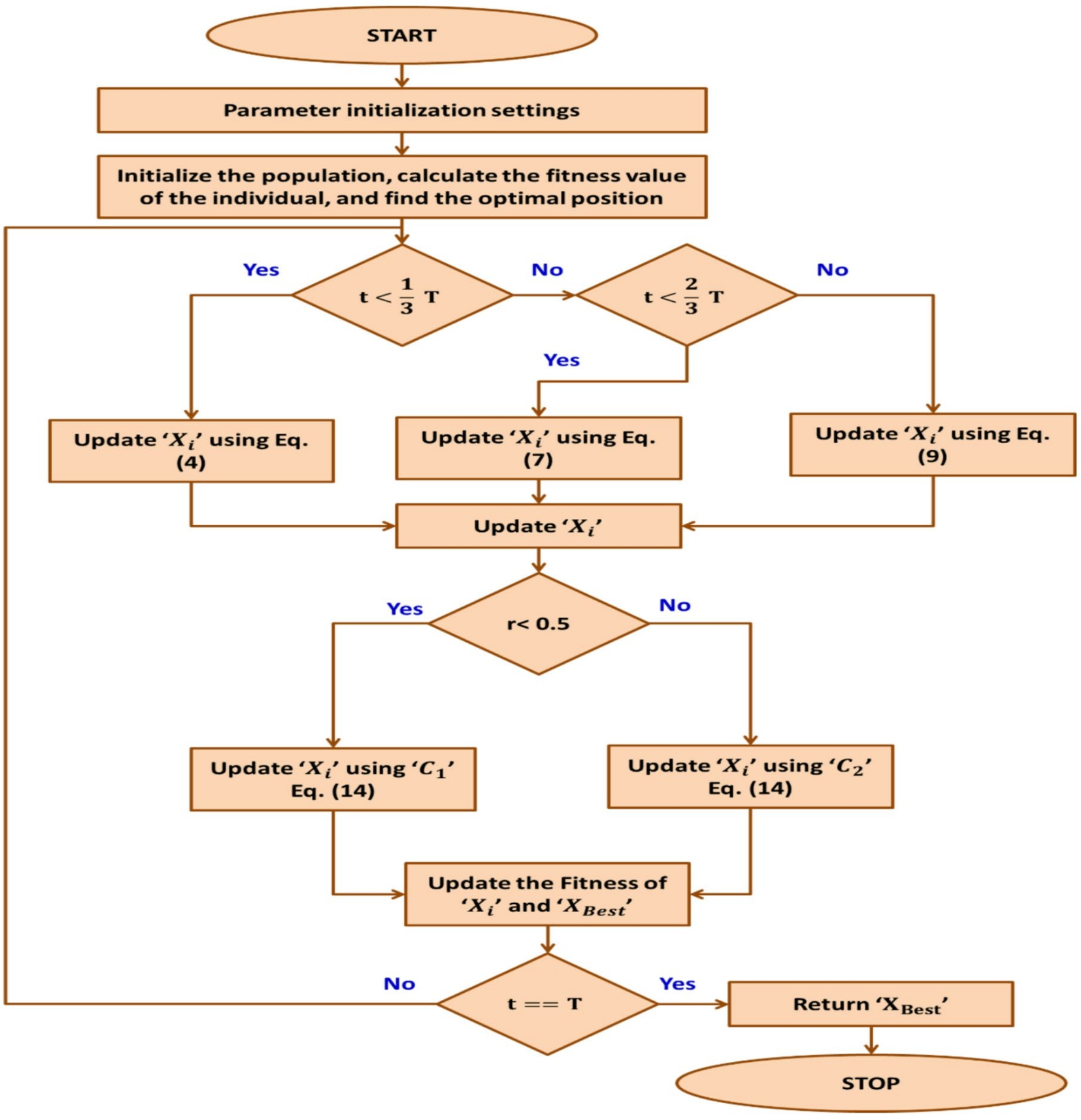
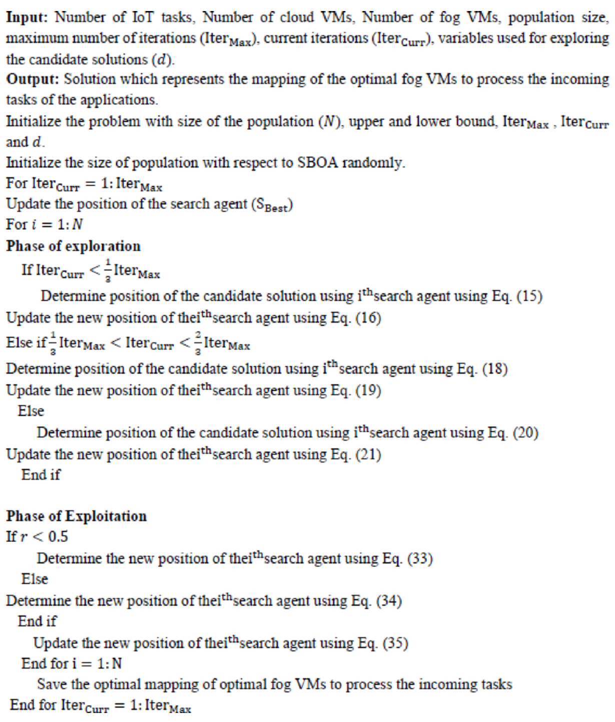

START
Parameter initialization settings
Initialize the population, calculate the fitness value of the individual, and find the optimal position
Yes
$t < \frac{1}{3}T$
No
$t < \frac{2}{3}T$
No
Yes
Update 'X_i' using Eq. (4)
Update 'X_i' using Eq. (7)
Update 'X_i' using Eq. (9)
Update 'X_i'
Yes
r< 0.5
No
Update 'X_i' using 'C_1' Eq. (14)
Update 'X_i' using 'C_2' Eq. (14)
Update the Fitness of 'X_i' and 'X_{Best}'
No
t == T
Yes
Return 'X_{Best}'
STOP

Fig. 4. The proposed IMSESBOA-based IoT TS algorithm



```
Input: Number of IoT tasks, Number of cloud VMs, Number of fog VMs, population size,
maximum number of iterations ($\text{Iter}_{\text{Max}}$), current iterations ($\text{Iter}_{\text{Curr}}$), variables used for exploring
the candidate solutions ($d$).
Output: Solution which represents the mapping of the optimal fog VMs to process the incoming
tasks of the applications.
Initialize the problem with size of the population ($N$), upper and lower bound, $\text{Iter}_{\text{Max}}$ , $\text{Iter}_{\text{Curr}}$
and $d$.
Initialize the size of population with respect to SBOA randomly.
For $\text{Iter}_{\text{Curr}} = 1{:}\,\text{Iter}_{\text{Max}}$
Update the position of the search agent ($S_{\text{Best}}$)
For $i = 1{:}\,N$
Phase of exploration
  If $\text{Iter}_{\text{Curr}} < \frac{1}{3}\text{Iter}_{\text{Max}}$
     Determine position of the candidate solution using $\text{i}^{\text{th}}$search agent using Eq. (15)
Update the new position of the$\text{i}^{\text{th}}$search agent using Eq. (16)
Else if$\frac{1}{3}\text{Iter}_{\text{Max}} < \text{Iter}_{\text{Curr}} < \frac{2}{3}\text{Iter}_{\text{Max}}$
Determine position of the candidate solution using $\text{i}^{\text{th}}$search agent using Eq. (18)
Update the new position of the$\text{i}^{\text{th}}$search agent using Eq. (19)
  Else
     Determine position of the candidate solution using $\text{i}^{\text{th}}$search agent using Eq. (20)
Update the new position of the$\text{i}^{\text{th}}$search agent using Eq. (21)
  End if

Phase of Exploitation
If $r < 0.5$
     Determine the new position of the$\text{i}^{\text{th}}$search agent using Eq. (33)
  Else
Determine the new position of the$\text{i}^{\text{th}}$search agent using Eq. (34)
  End if
     Update the new position of the$\text{i}^{\text{th}}$search agent using Eq. (35)
  End for $\text{i} = 1{:}\,\text{N}$
     Save the optimal mapping of optimal fog VMs to process the incoming tasks
 End for $\text{Iter}_{\text{Curr}} = 1{:}\,\text{Iter}_{\text{Max}}$
```

Algorithm 1: Implementation of the proposed IMSESBOA-based IoT TS algorithm.

In addition, Algorithm 1 presents the Implementation steps included into the proposed IMSESBOA-based IoT TS algorithm for determining optimal schedules in fog-cloud computing scenario.

Based on the above-mentioned IMSESBOA-based IoT TS algorithm, the tasks scheduler selected the best and most optimal VMs associated with the fog and cloud for executing the IoT tasks on the user side. It specifically derived the benefits of CSBOA with multiple number of strategies which has good strength in terms of attaining shortened makespan and minimized execution time, it also possessed a good strength in determining near-optimal local solution with maximized accuracy.

Reinforcement learning (RL) technique

The merits of Q-learning algorithm-based RL are used for updating the position of the individual in the population[35,36]. RL is specifically employed to characterize and address the problems through which the agents acquire different methodologies to maximize returns[37,38]. These returns aid in establishing the required objectives using interactions with the environment. It is used for learning how well the mapping between the state and the action can be achieved for the purpose of maximizing the reward. The agents under RL select an action and employ a specific strategy for execution depending on the current state realized in the environment. When the agent's action is received in the environment, the status is updated such that the agent receives reward feedback. In specific, Q-learning pertains to the value-based algorithm associated with RL. It is a model free-type machine leaning approach in which the agent does not require the necessity of understanding the environment. It uses states and actions for a specific environment, in which state refers to the observed value and the sample determined from the environment. On the other hand, the action is the selected option made by the agent after the process of observing the environment. At this juncture, environment assigns a reward based on determined state, and the primitive concept of Q-learning takes the responsibility of constructing a Q-Table which comprises of reserved Q-values which is associated with each pair of state and action identified from the environment.

$$Q_{i+1}\left(S_i, a_i\right) = Q_i\left(S_i, a_i\right) + \alpha\ *\delta \tag{36}$$

Where $Q_i\left(S_i, a_i\right)$and $Q_{i+1}\left(S_i, a_i\right)$represents the expectation that an action is achieved based on the facilitation of a specific state at time t, and the Q-value which is rewarded for making the agent to learn about the environment after the specific time $t+1$. In specific δ and α represents the estimation error and the learning rate which ranges between 0 and 1. In this context, the value of δ is determined based on Eq. (37)

$$\delta\ = r_{act(t+1)} + ds_{coeff[0,1]} \cdot Q_{i+1}\left(S_i, a_i\right) - Q_i\left(S_i, a_i\right) \tag{37}$$

However, the value of estimation error decreases as it converges towards the target. Further $ds_{coeff[0,1]}$ and $r_{act(t+1)}$ represents the discount coefficient which lies between 0 and 1, and the reward achieved after performing an action a_i in the state S_i. This $r_{act(t+1)}$ reward is considered to be finite during the process of continuous learning.

Integration of RL into IMSESBOA approach

The IMSESBOA approach incorporates uniform updating operations across all individuals in the population, wherein each individual is iteratively refined through the exploration and exploitation phases. Transition between phases of exploration and exploitation is essential for attaining a better balance between exploration and exploitation rates such that algorithmic performance is improved during the process of optimization. At the same time, the RL method possesses the capability of learning from the tasks and environments in an adaptive manner. It also helps in determining better solutions by adjusting the strategies depending on the results of the learning process. This potentiality offered by RL helps in providing flexibility under the influence of unknown, dynamic and complex environments. Thus, the concept of RL is utilized for setting reasonable rewards after the process of updating and supporting the individuals of the population. These merits of RL attribute towards the determination of the solutions which are suitable for updating phases during the process of optimization. Then the method of fitness evaluation is used after exploration and exploitation phases for retaining the better individuals in the population. But the inclusion of RL into IMSESBOA helps to evaluate the fitness value only once. This combination effectively mitigates premature convergence and prevents the solution from becoming trapped in local optima during successive search iterations. In specific, the phases of RL- IMSESBOA helps in updating the phase of prey attacking (specified in Eq. (25)) as specified in Eq. (38)

$$\mathrm{C_{SB}}_{(\mathrm{i})}^{\mathrm{PA-New\ (t)}} = \mathrm{C_{SB(i)}}\,(\mathrm{t}) + \left(\mathrm{C_{SB(r1)}}\,(\mathrm{t}) - TF_{Mean}\right) \times \mathrm{R_{V(1)}} \tag{38}$$

Where $\mathrm{C_{SB(i)}}\,(\mathrm{t})$ and $\mathrm{R_{V(1)}}$is the individual of the population with the best fitness value and the random number which ranges between 0 and 1. In specific, $TF_{Mean}$is the random integer which lies between 1 and 2 with $Mean$ as the average fitness value associated with the complete set of individuals in the population. Further a simple selection probability factor is proposed for computing the indices based on Eq. (39)

$$\mathrm{SP_{Index}} = \frac{\mathrm{NI_{Pop}} - \mathrm{IN_{(I)}}}{\mathrm{NI_{Pop}}} \tag{39}$$

Where $\mathrm{NI_{Pop}}$and $\mathrm{IN_{(I)}}$represents the number of individuals in the population and the remaining individuals are determined after reordering the fitness values in ascending order.

Results and discussions

In this section, different experiments are conducted with the proposed IMSESBOA + RL and the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches with respect to the performance metrics of makespan, execution time, fitness cost and energy consumptions for varying number of IoT tasks. The complete experimentation is conducted using MATLAB R2018a which is installed over the system which comprises of Windows 10, 4 GB of RAM, Intel Core i5-4200U 2.4 GHz CPU. The experiments are conducted with different numbers of cloud and fog VMs that include (i) 5 Cloud VMs and 10 Fog VMs, (ii) 10 Cloud VMs and 20 Fog VMs and (iii) 15 Cloud VMs and 30 Fog VMs. Comprehensive set of experiments are conducted using the same machine which possesses the same characteristics. However, the fog and clouds in the implementation environment are considered to possess different processing power which is measured in terms of resource usage costs (in Grid Dollars (G$), Values of Bandwidth (Megabytes Per Second (MBPS), memory capacity (Megabytes (MB)) and processing power values (MIPS). In specific, G$ represents a unit of currency which is an alternate for real money through a predefined ratio[36]. This is considered as the virtual cost can be easily mapped to any of the pricing models considered for assessment. In addition, Table 1 lists the parameters used in the implementation of the proposed IMSESBOA scheme.

Three VM configurations (5 C-10 F, 10 C-20 F, 15 C-30 F) are selected as they are sufficient for exploring the potential of the proposed IMSESBOA + RL and the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches with respect to the scheduling process which targeted on the process of mapping to the maximum of 4000 IoT tasks incoming into the cloud-fog environment. The size of the population is 200 for the proposed IMSESBOA + RL and the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches for ensuring fair comparisons. In addition, 50 iterations are considered such that the mean value of the performance metrics is utilized for plotting the graphs in the subsequent sections.

Performance evaluation of the proposed IMSESBOA using Makespan for varying number of IoT tasks

In this first fold of analysis, the proposed IMSESBOA + RL and the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches are compared using makespan (seconds) for varying number of IoT tasks under three scenarios (5 Cloud VMs and 10 Fog VMs,10 Cloud VMs and 20 Fog VMs and 15 Cloud VMs and 30 Fog VMs) respectively. Figures 5, 6 and 7 demonstrate the plots of makespan which is defined as the total time utilized for executing the complete set of tasks and scheduling them to the processors for varying number of IoT tasks and three scenarios considered for evaluation.

This investigation with respect to makespan is conducted for the objective of identifying how the proposed IMSESBOA + RL approach is predominant over the baseline approaches in executing the complete set of tasks in the cloud-fog system. When the number of IoT tasks increases in the cloud-fog environment, the makespan increases since the time to execute the number of incoming tasks increase systematically. But this proportionally gets decreased with the increase in the number of clouds VMs and fog VMs. Since the resource used for allocating and executing the scalable number of IoT tasks increases in the cloud-fog computing system. But the proposed IMSESBOA + RL approach is capable in executing the tasks with the necessitated deadline time such

Parameter	Unit	Value
Characteristics of real time IoT tasks		
Time of Broker Arrival	Second	1–100
Size of input file	MB	10–100
Size of output file	MB	10–100
Memory	MB	10–150
Task length	MIPS	1000–20,000
Characteristics of cloud VMs		
Cost of bandwidth utilization	G$/MB	0.06–0.1
Cost of memory utilization	G$/MB	0.03–0.06
Cost of CPU utilization	G$/Second	0.6–1.0.6.0
Bandwidth	Mbps	512–4096
RAM	MB	5000–20,000
Computing power	MIPS	3000–5000
Number of VMs	VM	5,10,15
Characteristics of fog VMs		
Cost of bandwidth utilization	G$/MB	0.01–0.03
Cost of memory utilization	G$/MB	0.01–0.04
Cost of CPU utilization	G$/Second	0.2–0.5
Bandwidth	Mbps	128–1024
RAM	MB	250–5000
Computing power	MIPS	1000–2000
Number of VMs	VM	10,20,30

Table 1. Parameters used in the implementation of the proposed IMSESBOA scheme

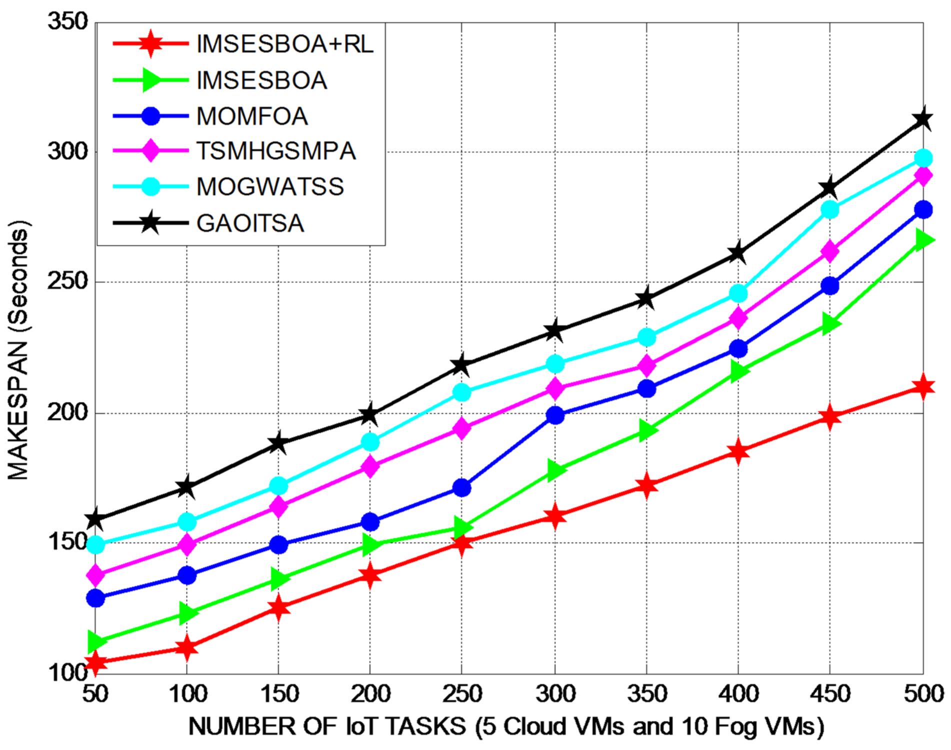


Fig. 5. Proposed IMSESBOA-Makespan for varying number of IoT Tasks (5 Cloud VMs and 10 Fog VMs)

that makespan is significantly decreased during the TS process. This excellence of the proposed IMSESBOA + RL approach is mainly due to the multi-strategy and RL included during process of exploration and exploitation that aids in mapping the IoT tasks to the available fog and cloud VMs. The adoption of multi-strategies and RL into the traditional SBOA helped in supporting the actions of the agents for balancing the tradeoff between makespan

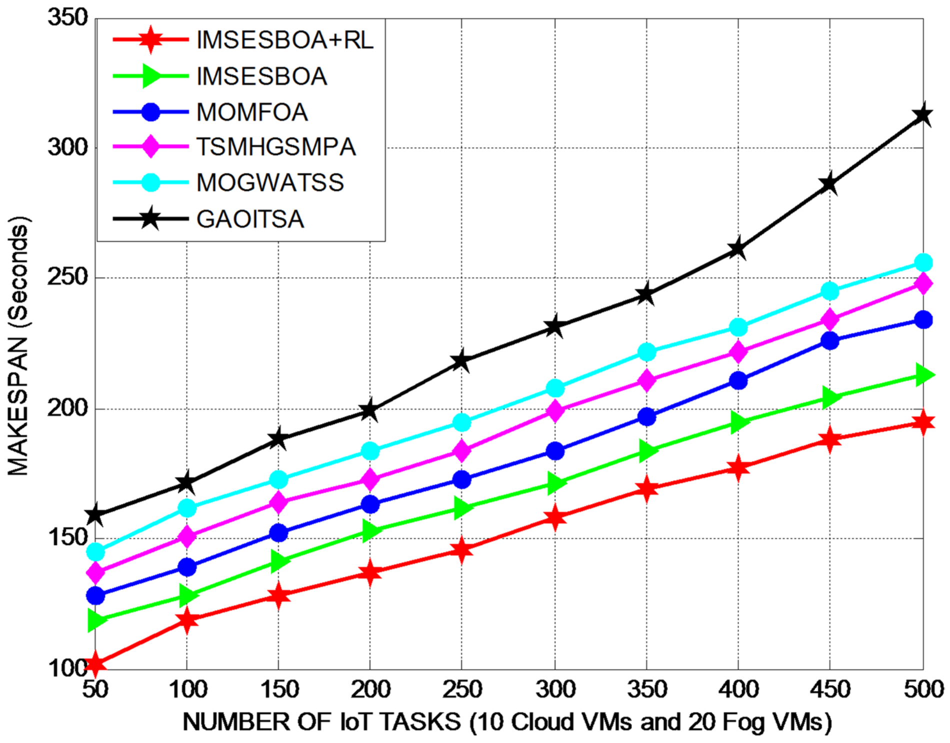


Fig. 6. Proposed IMSESBOA-Makespan for varying number of IoT Tasks (10 Cloud VMs and 20 Fog VMs)

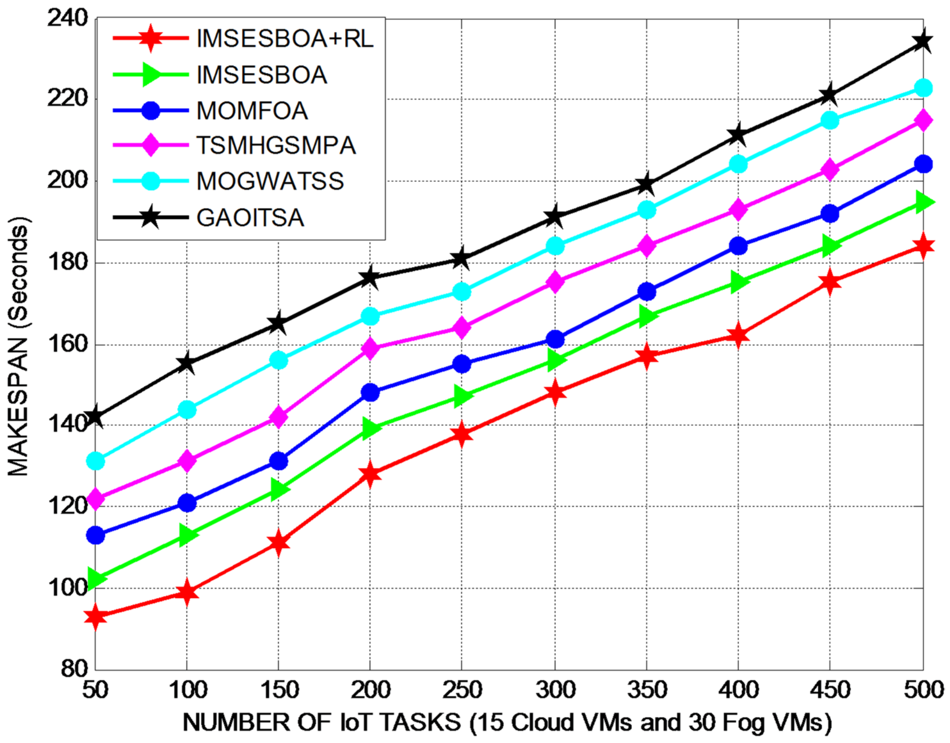


Fig. 7. Proposed IMSESBOA-Makespan for varying number of IoT Tasks (15 Cloud VMs and 30 Fog VMs)

and execution with maximized solution diversity and proper utilization of global information shared in the scenario. On the other hand, MOMFOA utilized multi-objective MFOA algorithm for well placing the tasks to the fog and cloud VMs with confirmed objectives of QoS in the network. This IMSESBOA + RL approach well balanced the resources of the fog-cloud environment with respect to the tasks which eventually gets increased in the environment with corresponding increase in the makespan. However, the proposed IMSESBOA + RL approach included a CC strategy which helps the search agents to work together such that they cooperatively balance exploration and exploitation rates such that optimal solutions are determined at a rapid rate without any delayed convergence. It further plays a crucial role in avoiding entrapment in local optima, which in turn leads to significant reductions in energy consumption and cost during IoT task execution within real-world implementation scenarios. It also facilitated a balanced search such that potential regions are only exploited during the process of IoT tasks migration under the execution scenario.

On the other hand, the baseline TSMHGSMPA and MOGWATSS are hybrid approaches of TS, but well balance between the local and global searching is not realized in the implementation scenario. Moreover, the benchmarked GAOITSA approach possessed the limitations of poor solution diversity and ignored the use of global information shared between the entities in cloud-fog system. As in Fig. 5, With respect to 5 Cloud VMs and 10 Fog VMs, the proposed IMSESBOA + RL approach with different IoT tasks minimized the makespan by 13.28%, 15.56%, 16.52%, 17.64% and 18.56% better than the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches. On the other hand, the proposed IMSESBOA + RL approach with respect to 10 Cloud VMs and 20 Fog VMs with different IoT tasks reduced the makespan by 12.68%, 14.76%, 15.45%, 16.92% and 17.62%, better than the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches as presented in Fig. 6. In addition, the proposed IMSESBOA + RL approach with respect to 15 Cloud VMs and 30 Fog VMs with different IoT tasks reduced the makespan on average by 12.45%, 14.76%, 15.24%, 16.42% and 17.54% better than the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches as shown in Fig. 7.

Performance evaluation of the proposed IMSESBOA using execution time for varying number of IoT tasks

In the second part of the investigation, the execution time (G$) incurred by the proposed IMSESBOA + RL and the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches are evaluated for varying number of IoT tasks under three scenarios that comprises of 5 Cloud VMs and 10 Fog VMs, 10 Cloud VMs and 20 Fog VMs and 15 Cloud VMs and 30 Fog VMs. This experiment is conducted using execution time to determine how the proposed IMSESBOA scheme is better than the baseline approaches in reducing the time of waiting in the queue, getting allocated to the resources and response for the initiated tasks is achieved in the cloud-fog system. This execution time completely depends on size of file considered as input, bandwidth, CPU and memory availability in the cloud-fog system. Thus, a reliable IoT TS approach needs to efficiently map the tasks to the existing resources such that QoS is highly satisfied with minimized delay in the network. The results of execution time presented in Figs. 8, 9 and 10 confirmed that the proposed IMSESBOA + RL approach is capable enough in facilitating a contextual alteration of satisfying the demands IoT tasks in the cloud-fog system.

The proposed IMSESBOA + RL approach specifically used the merits of RL and FR strategy for determining the possible mapping of tasks to the resources depending on the refinement imposed over the current state.

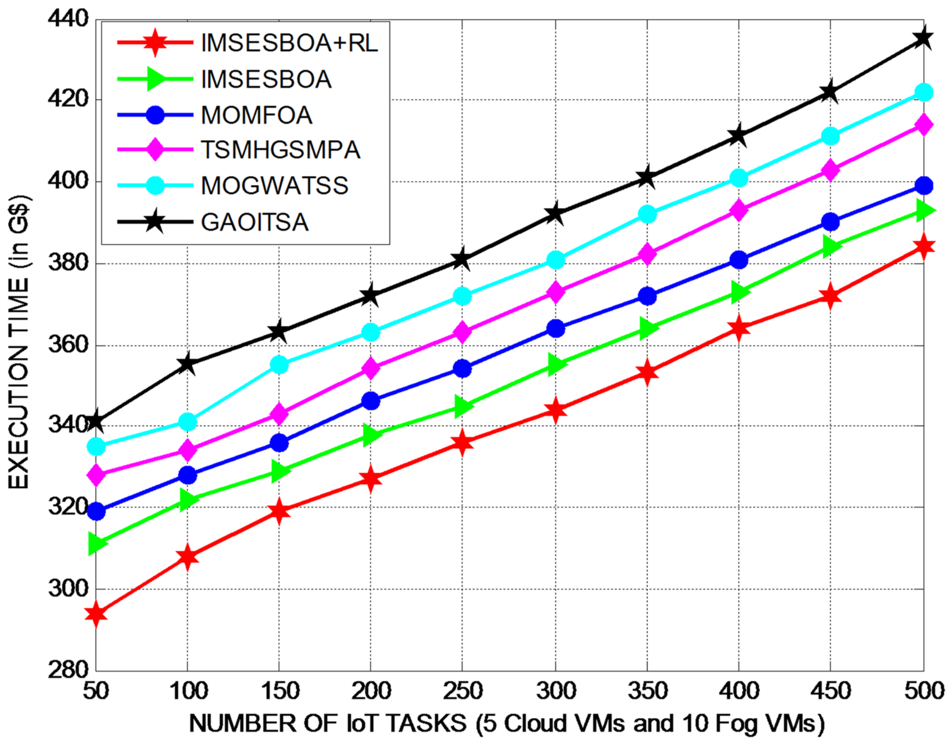


Fig. 8. Proposed IMSESBOA-Execution Time for varying number of IoT Tasks (5 Cloud VMs and 10 Fog VMs)

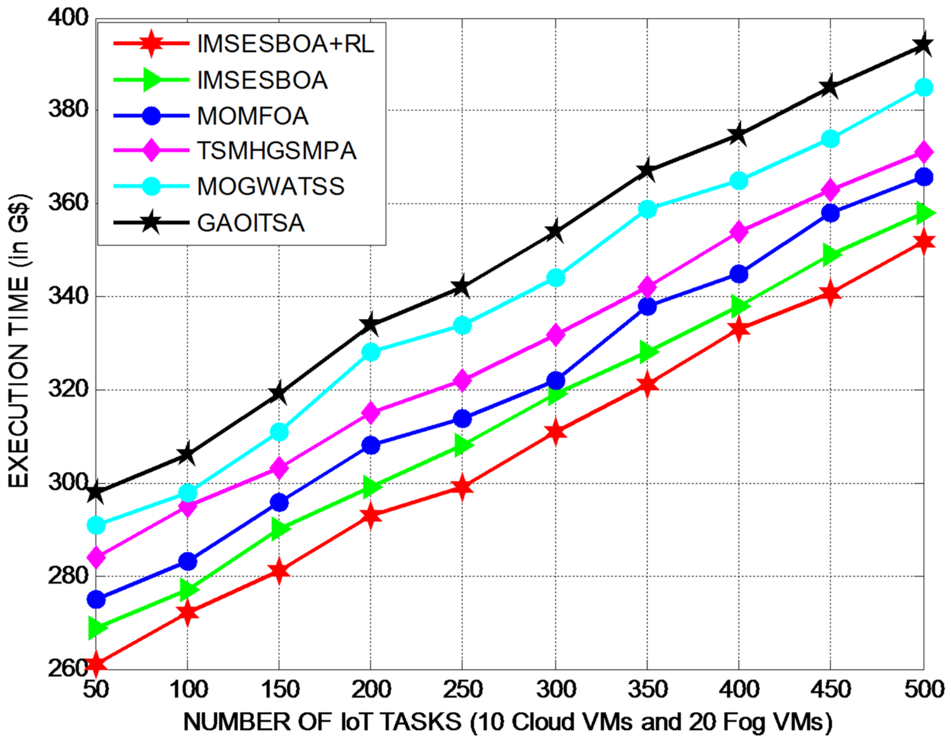


Fig. 9. Proposed IMSESBOA- Execution Time for varying number of IoT Tasks (10 Cloud VMs and 20 Fog VMs)

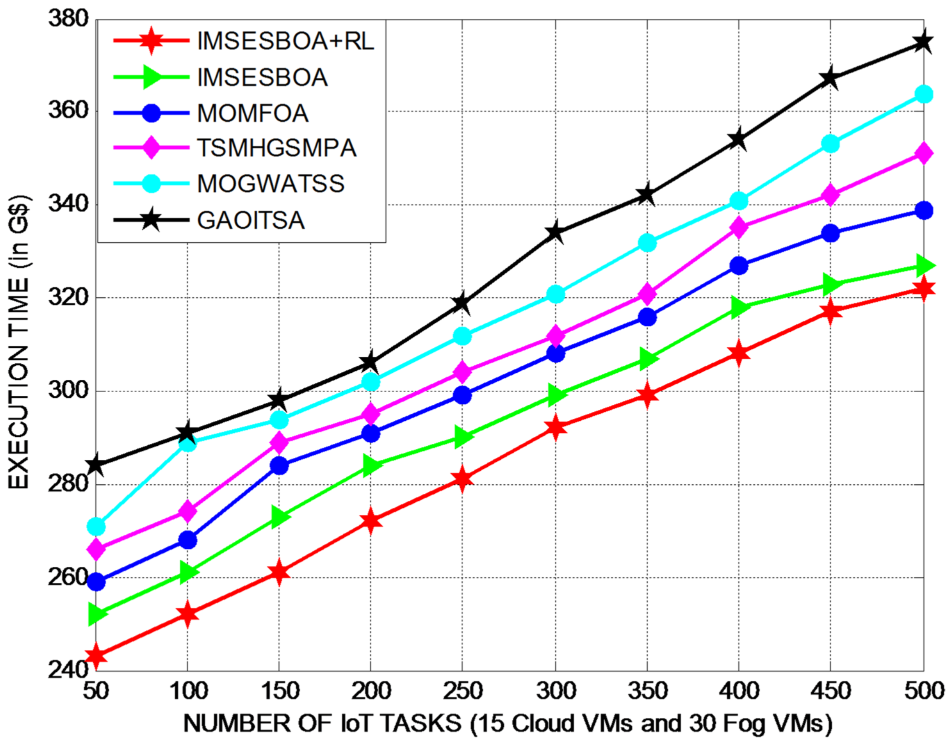


Fig. 10. Proposed IMSESBOA- Execution Time for varying number of IoT Tasks (15 Cloud VMs and 30 Fog VMs)

This blend of RL and FR approach included into the CSBOA algorithm played an anchor role in exchanging global information between the search agents such that each impact of the emerging factors is handled in a more reactive manner. It also used the benefits of CS strategy through the crowding distance can be computed for facilitating better mapping of tasks to the available fog and cloud VMs. This strategy of CS updating is used for determining how the current solution is changed compared to the previous solution in terms of magnitude such that the positions of the current solution need to be replaced are identified. It helps in determining the degree to which the current solutions' position is updated in the SS.

At the same time, MOMFOA and TSMHGSMPA approaches ignored the inclusion of reactive factors that attributed towards better resource mapping processes. They were also marginal in performing dynamic adjustment since they contributed as a semi-dynamic approach of TS. At the same time, MOGWATSS and GAOITSA included only a limited number of factors which helped in mapping of tasks to existing VMs of the cloud-fog environment. Thus, the proposed IMSESBOA + RL approach in the presence of 5 Cloud VMs and 10 Fog VMs, minimized the execution time by 13.32%, 16.54%, 17.18%, 18.76% and 19.36%, better than the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches as shown in Fig. 8. This proposed IMSESBOA + RL approach with respect to 10 Cloud VMs and 20 Fog VMs further reduced the execution time on average by 11.65%, 13.76%, 14.42%, 15.69% and 16.54%, better than the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches as detailed in Fig. 9. In addition, the proposed IMSESBOA + RL approach with respect to 15 Cloud VMs and 30 Fog VMs and IoT tasks also reduced the execution time on average by 11.76%, 13.86%, 14.54%, 15.76% and 16.64% better than the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches as presented in Fig. 10.

Performance evaluation of the proposed IMSESBOA using fitness function

In this section, the fitness function used for TS in the cloud-fog computing environment is investigated for varying number of tasks. This fitness function is used for establishing a better balance between the total execution cost and makespan with respect to the real-time tasks entering the cloud-fog computing scenario. This problem of IoT TS is developed as a maximizing optimization problem, and hence the high fitness value infers a better balance between the execution cost and makespan. This experiment is mainly conducted for assessing and realizing the degree of balance existing between the proposed IMSESBOA + RL and the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches for varying number of IoT tasks under 10 Cloud VMs and 20 Fog VMs, and 15 Cloud VMs and 30 Fog VMs as depicted in Figs. 11 and 12 respectively.

The above plots clearly proved that the proposed IMSESBOA approach with respect to scalable increase in the number of IoT tasks performed well on par with the baseline approaches since it established a better tradeoff between the total execution cost and makespan. The potentiality of this proposed IMSESBOA approach is mainly due to the incorporation of the GSG method which wide opened the option of integrating the merits of golden section search which is a numerical method which is helpful in identifying the minimum of the function. It targeted the process of enhancing the potential of the algorithm such that potential regions of the SS are explored such that it guides towards the process of determining an optimal solution.

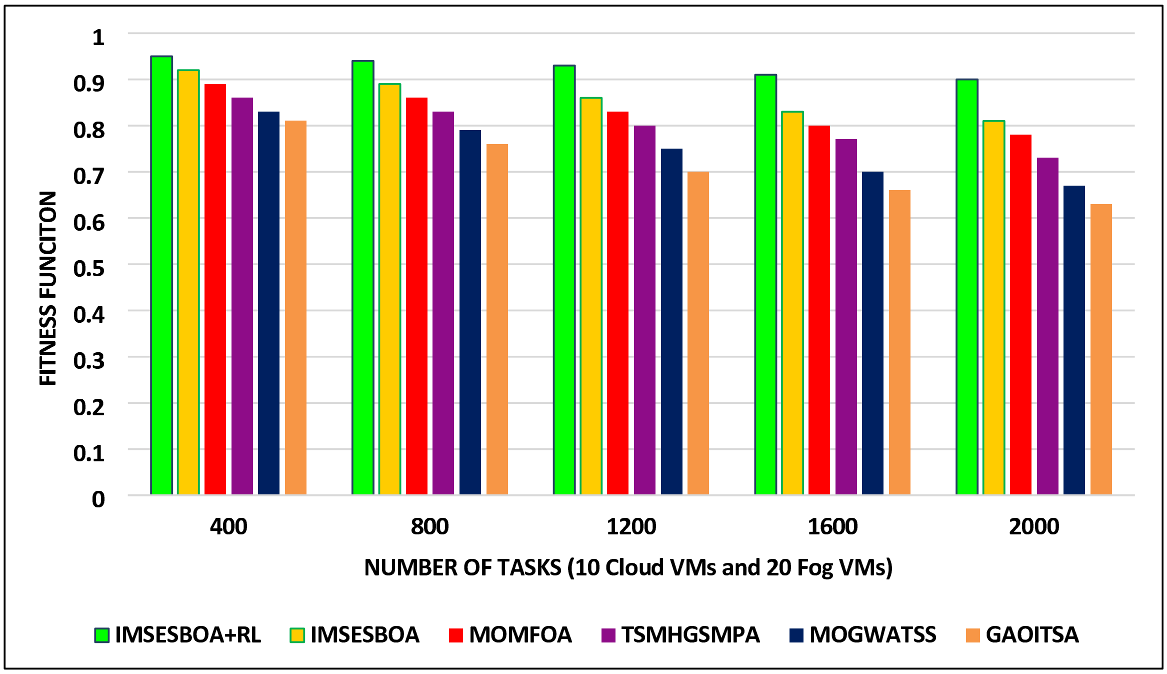


Fig. 11. Proposed IMSESBOA-Fitness Function value for varying number of tasks during IoT TS(10 Cloud VMs and 20 Fog VMs)

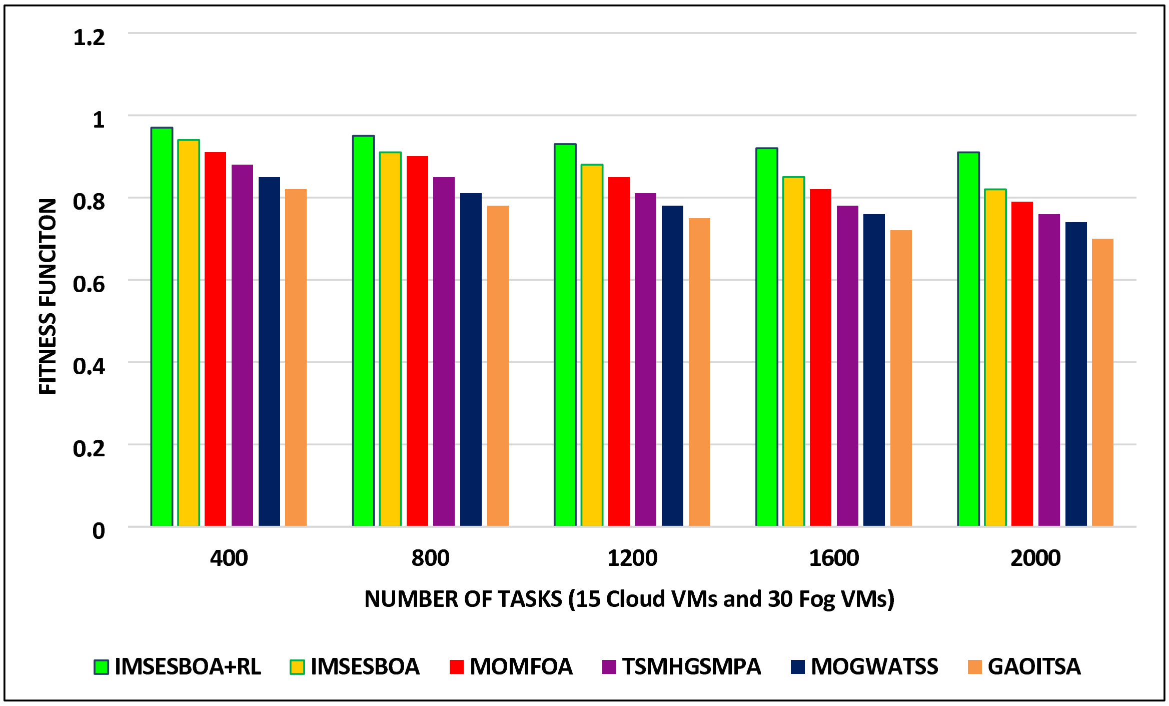


Fig. 12. Proposed IMSESBOA-Fitness Function value for varying number of tasks during IoT TS (15 Cloud VMs and 30 Fog VMs)

At the same time, the baseline MOMFOA and TSMHGSMPA approaches exhibited a similar kind of pattern, but this trend decreased with large increase in the number of IoT tasks in the cloud-fog computing scenario. This is because, these approaches could not balance the impact of large tasks in a reactive way by adopting to the influence parameters of TS. Likewise, the other benchmarked MOGWATSS and GAOITSA even through being stochastic in nature could not perform well with a well balance since the probability of offloading was computed randomly without considering the factors of uncertainty into account. But the proposed IMSESBOA + RL approach confirmed the potential of efficiently exploring the large SS as it adopted multi-strategies into the classical SBOA for attaining balance between the phases of exploration and exploitation and reinforcement leaning useful for learning patterns from new workloads.

Performance evaluation of the proposed IMSESBOA approach using energy consumptions and cost with scalable number of IoT tasks

In this section, the energy consumption and cost incurred by proposed IMSESBOA + RL and the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA schemes with scalable number of IoT tasks are compared and the results are presented. This comparative investigation is conducted using energy consumption and cost for the objective of determining how well the proposed IMSESBOA approach and the baseline schemes perform in the real-world IoT applications where energy constraints are critical. In specific, Results in Figs. 13 and 14 demonstrates the plots of energy consumption incurred by the IMSESBOA + RL approach and the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches under 1000 to 5000 tasks with 10 cloud VMs and 20 Fog VMs and 15 cloud VMs and 30 Fog VMs. In both the scenarios, the proposed IMSESBOA approach exhibited better performance since the adoption of multi-strategy enhanced SBOA explored and exploited the SS for mapping incoming IoT tasks to the suitable VMs for rapid execution which considerably minimized the energy spent during the execution process. This excellence of the proposed IMSESBOA approach under scalable number of IoT tasks is evidently realized during implementation since the deadline constraints considered during the optimization process is highly dynamic and ideally suits during the process of IoT tasks execution with comparatively minimized delay. These reactive deadline constraints were not utilized by the baseline MOMFOA and TSMHGSMPA approaches, and hence they performed marginally compared to proposed IMSESBOA approach. Tthe thresholds considered during the assignment of fog resources to the incoming IoT tasks are not dynamic and hence possess a scope of improvement in terms of energy consumptions. Thus, the proposed IMSESBOA + RL approach minimized the energy consumption under 1000 to 5000 tasks with 10 cloud VMs and 20 Fog VMs by 7.16%, 8.12%, 10.54%, 12.31% and 13.98%, compared to the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches taken for comparison as in Fig. 13. Likewise, proposed IMSESBOA + RL approach minimized the energy consumption under 1000 to 5000 tasks with 15 cloud VMs and 30 Fog VMs by 6.78%, 7.84%, 9.42%, 11.76% and 12.64%, compared to the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches used for comparison as depicted in Fig. 14.

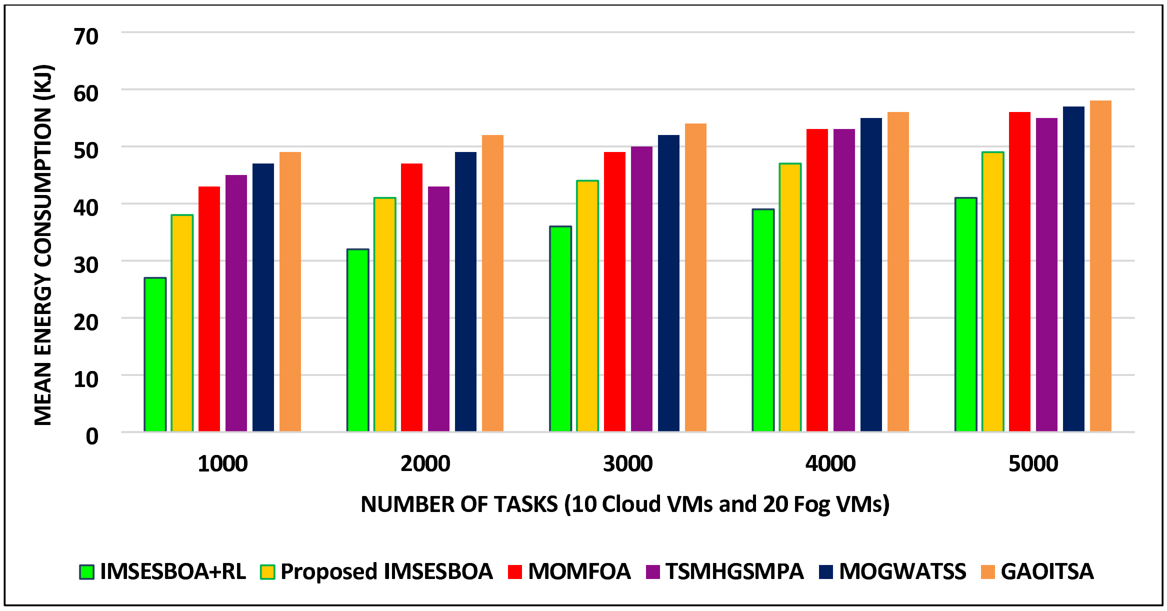


Fig. 13. Proposed IMSESBOA + RL-Mean Energy Consumptions for varying number of tasks during IoT TS(10 Cloud VMs and 20 Fog VMs)

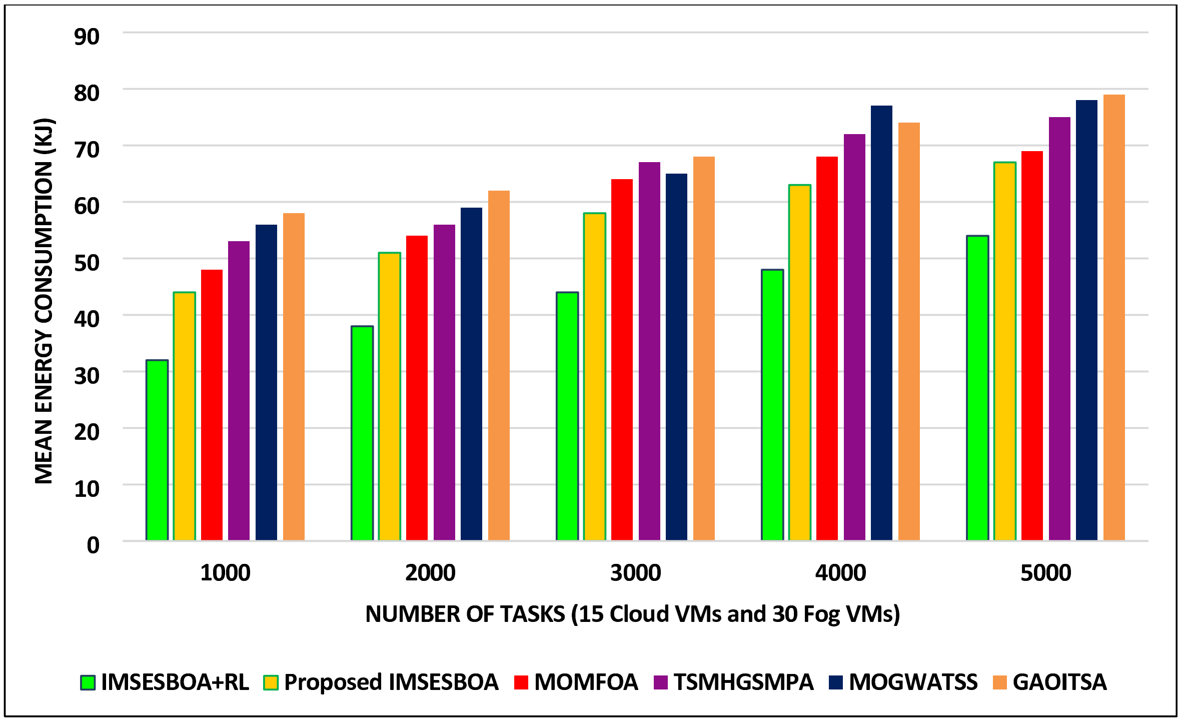


Fig. 14. Proposed IMSESBOA + RL-MeanEnergy Consumptions for varying number of tasks during IoT TS (15 Cloud VMs and 30 Fog VMs)

Furthermore, the plots of mean cost incurred by the IMSESBOA + RL approach and the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches under 1000 to 5000 tasks with 10 cloud VMs and 20 Fog VMs and 15 cloud VMs and 30 Fog VMs. Thus, the proposed IMSESBOA + RL approach minimized the mean cost under 1000 to 5000 tasks with 10 cloud VMs and 20 Fog VMs by 7.16%, 8.12%, 10.54%, 12.31% and 13.98%, compared to the proposed IMSESBOA + RL and the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches used for comparison as shown in Figs. 15. Likewise, the proposed IMSESBOA + RL minimized the mean cost under 1000 to 5000 tasks with 15 cloud VMs and 30

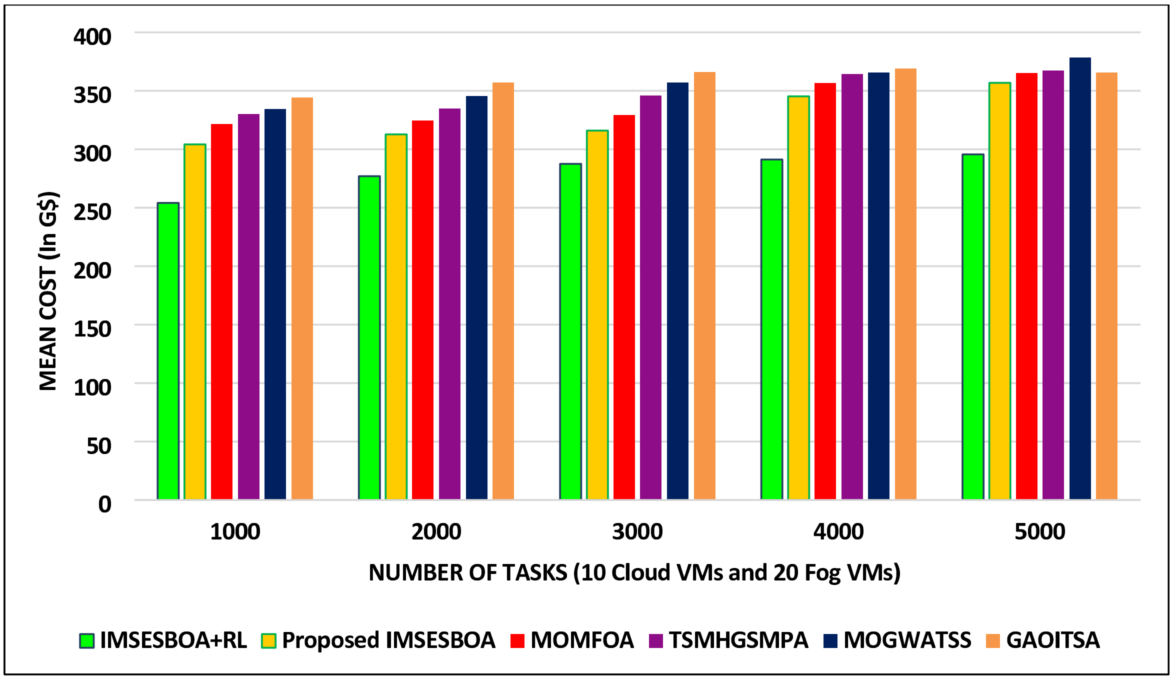


Fig. 15. Proposed IMSESBOA + RL-Mean Cost for varying number of tasks during IoT TS(10 Cloud VMs and 20 Fog VMs)

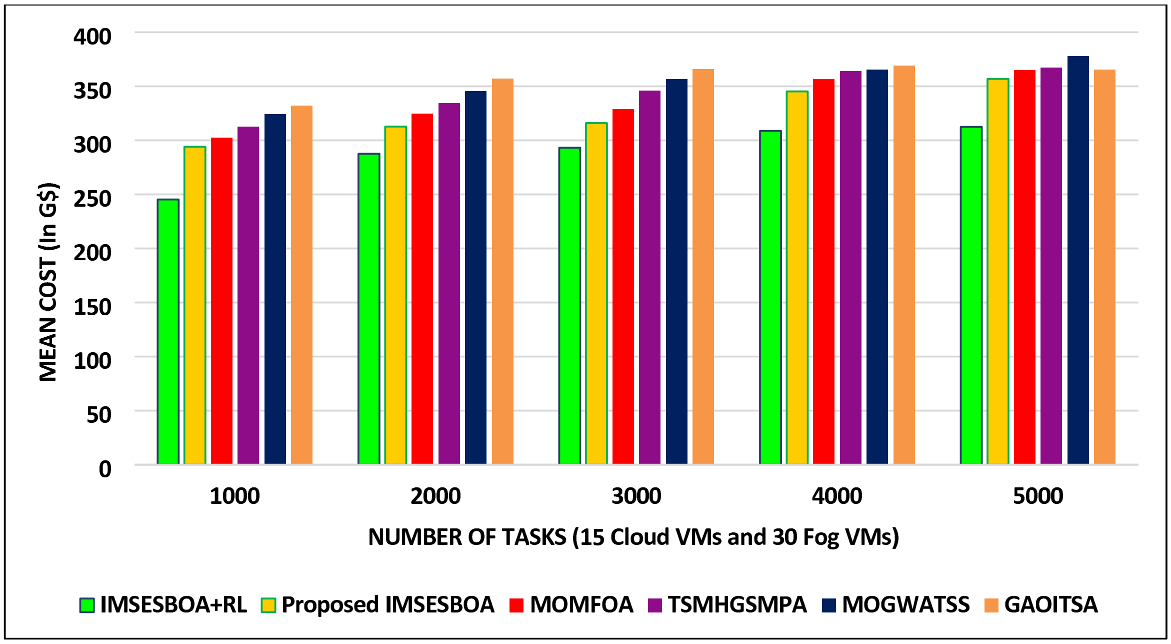


Fig. 16. Proposed IMSESBOA + RL-Mean Cost for varying number of tasks during IoT TS (15 Cloud VMs and 30 Fog VMs)

Fog VMs by 7.42%, 8.32%, 10.21%, 11.56% and 13.84%, compared to the proposed IMSESBOA and the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches used for comparison as in Figs. 16.

Figure 17 presents the scheduling overhead incurred by the proposed IMSESBOA + RL approach and the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches under different number of tasks which need to be scheduled in the fog computing environment. This result confirmed that the proposed IMSESBOA + RL approach reduced the scheduling overhead relatively well compared to the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches since it dynamically learnt

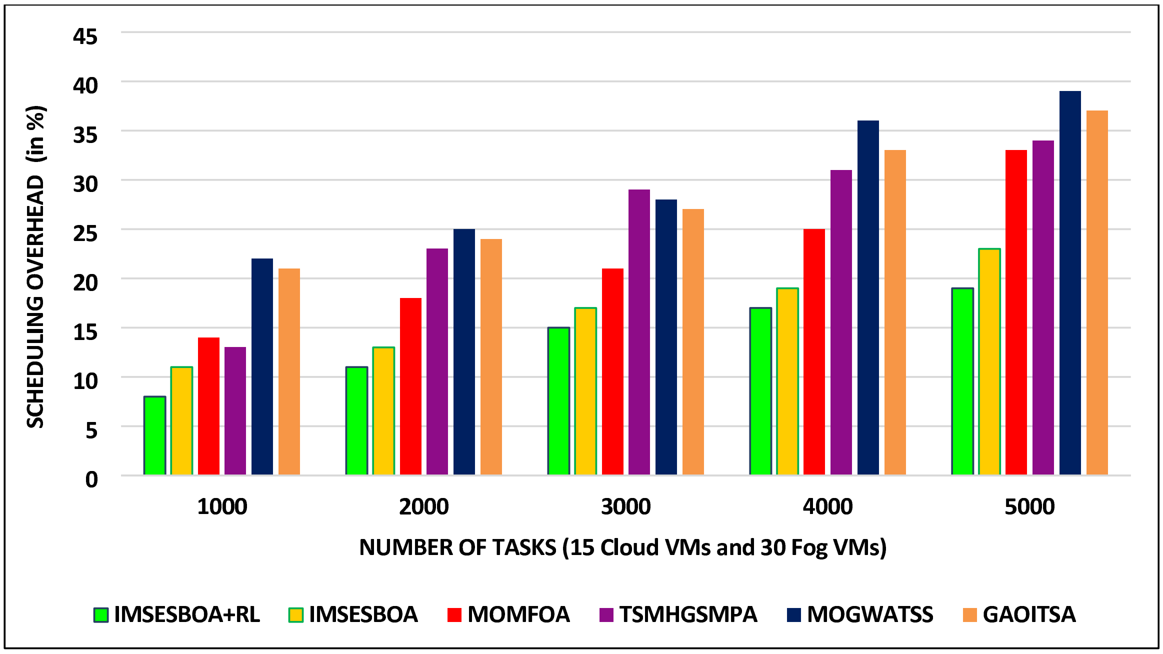


Fig. 17. Proposed IMSESBOA + RL-Mean Cost for varying number of tasks during IoT TS (15 Cloud VMs and 30 Fog VMs)

the patterns of the tasks under scheduling using the agents of RL which reactively understands the different features which decides upon the process of scheduling in the fog environment. In specific, RL is potent in dynamically adapting to the new workloads since it excels in learning optimal strategies using the interaction of error and trial with the environment. This use of error and trail interaction does not necessitate any pre-defined training data. This significance of RL is specifically useful in dynamic environments in which the situations are constantly changing, and the optimal solution is completely unknown. This inclusion of reinforcement leaning also permits the optimization associated with resource allocation and continues improvement in TS based on scalable increase in the number of tasks to be scheduled in the fog computing scenario. Hence the proposed IMSESBOA + RL minimized the scheduling cost under 1000 to 5000 tasks with 15 cloud VMs and 30 Fog VMs by 6.98%, 7.42%, 8.21%, 10.65% and 11.94% compared to the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches used for comparison.

In addition, Fig. 18 presents the computational overhead facilitated by the proposed IMSESBOA + RL approach and the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches under different number of tasks. It is seen that the computational overhead during the application of the approaches got increased with increase in the tasks. Even though the use of RL facilitated better performance on one side, it is also responsible for comparative increase in the computation overhead. The use of RL helped in enhancing the process of decision-making during TS by permitting an agent to learn the situations through trial and error. But the process of learning frequently necessitates essential computational resources for attaining maximized rewards. These training processes also handle vast state space and complex environment. Hence the proposed IMSESBOA + RL minimized the computational overhead under 1000 to 5000 tasks with 15 cloud VMs and 30 Fog VMs by 7.76%, 8.42%, 9.98%, and 11.21%, compared to the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches used for comparison. But in contrast, MOGWATSS performed comparatively well than the proposed IMSESBOA + RL approach by reducing the computational overhead by 5.76% under different number of tasks.

Results of ANOVA test

An ANOVA test is conducted to determine the potentiality of proposed IMSESBOA approach compared to IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA, respectively. The null and alternate hypothesis considered for this ANOVA statistical tests is listed as follows.

Null hypothesis ($\mathbf{H_0}$): There is no difference in makespan and execution time in fog-cloud system after scheduling resources to IoT application tasks using proposed IMSESBOA approach.

Alternate hypothesis ($\mathbf{H_1}$): There is a substantial difference in makespan and execution time in fog-cloud system after scheduling resources to IoT application tasks using proposed IMSESBOA approach.

In specific, Tables 2 and 3 presents the ANOVA test results of the proposed IMSESBOA scheme in terms of makespan and execution time.

From the above-mentioned ANOVA test results, it is evident that F-Statistic in both the vases is greater than the F critical value, and the value of p is smaller than 0.05 for the parameters of makespan and execution time. Hence, the null hypothesis (H_0) is rejected and Alternate Hypothesis (H_1) is accepted. Thus, there is a

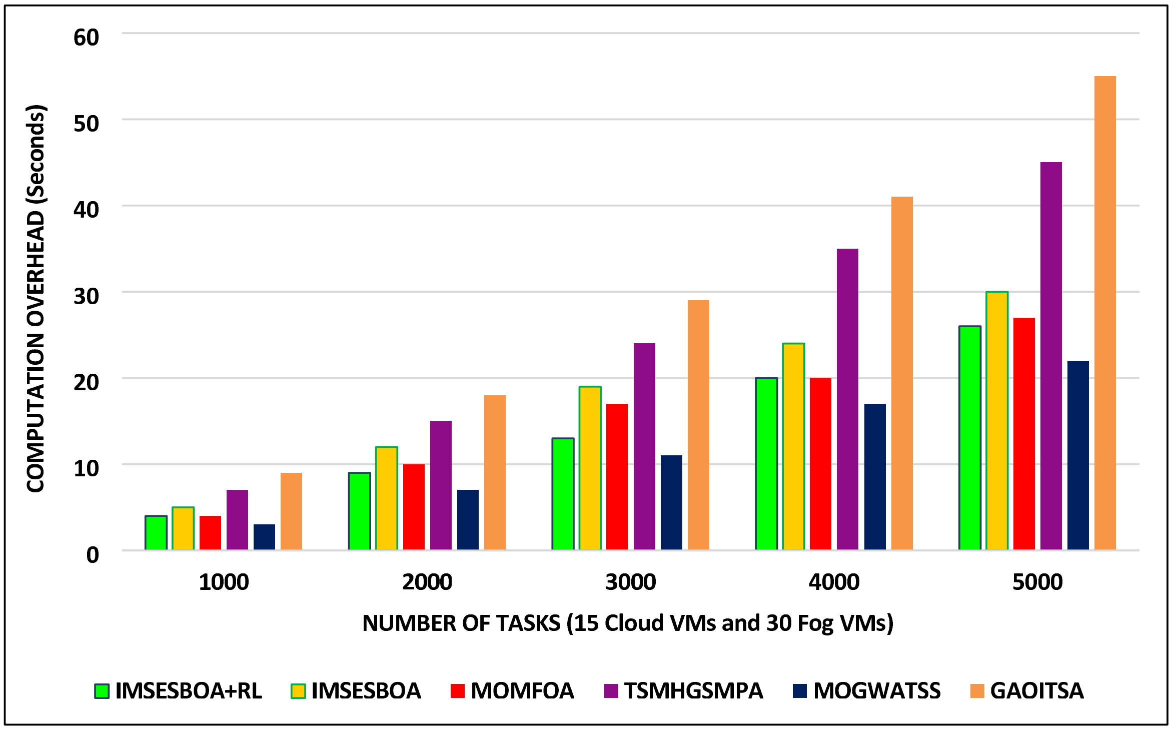


Fig. 18. Proposed IMSESBOA + RL-Computation Overhead Mean Cost for varying number of tasks during IoT TS (15 Cloud VMs and 30 Fog VMs)

Groups	**Count**	**Sum**	**Mean**	**Variance**		
Proposed IMSESBOA	20	1542	77.10	27.84		
MOMFOA	20	1436	72.80	25.92		
TSMHGSMPA	20	1032	51.60	19.61		
MOGWATSS	20	986	49.32	18.46		
GAOITSA	20	1154	57.72	21.98		
ANOVA						
Source of Variation	**SS**	**df**	**MS**	**F**	**P-Value**	**F-Critical**
Between Groups	4562.56	4	1140.64	224.094	3.12432E-21	3.2476
Within Groups	428	84	5.09			
Total	5124.86	99				

Table 2. ANOVA test results of the proposed IMSESBOA scheme with respect to Makespan

Groups	**Count**	**Sum**	**Mean**	**Variance**		
Proposed IMSESBOA	20	1896	94.86	27.45		
MOMFOA	20	1675	83,75	25.64		
TSMHGSMPA	20	1198	59.98	26.82		
MOGWATSS	20	1021	51.05	24.41		
GAOITSA	20	1118	55.98	22.68		
ANOVA						
Source of Variation	**SS**	**df**	**MS**	**F**	**P-Value**	**F-Critical**
Between Groups	4876.32	4	1219.06	239.26	3.0245E-21	3.2186
Within Groups	428	84	5.095			
Total	5453.94	99				

Table 3. ANOVA test results of the proposed IMSESBOA scheme with respect to execution Time

significant difference in the makespan and execution time in the fog-cloud system after the process of scheduling the resources to the IoT application tasks using the proposed IMSESBOA approach.

Computational complexity

Let us consider the number of total IoT tasks which need to be scheduled over m number of available VMs be n with k as the number of dependency constraints imposed during the process of resource allocation in the fog-cloud environment. Then the time complexity of this scheduling algorithms completely depends on the computation incurred for carrying out the necessitated essential operations such as allocation of suitable VMs to the IoT tasks and executing the allocated tasks which incurred the complexity of $\mathrm{O}\left(\mathrm{n}^{*}\mathrm{m}\right)$.Sorting the tasks during the process of execution incurs the cost of $\mathrm{O}\left(\mathrm{k}\right)$. Then the total computational complexity is $\mathrm{O}\left((\mathrm{n}^{*}\mathrm{m}) + \mathrm{n}(\mathrm{m}+\mathrm{k}+\mathrm{nk}\right)$ Thius the complexity of the proposed IMSESBOA approach is $\mathrm{O}\left(\mathrm{n}^{2*}\mathrm{k}\right)$.

Conclusions

The proposed IMSESBOA + RL-based IoT TS mechanisms during its application minimized the time incurred during data processing such that QoS is improved in fog-cloud computing. This IMSESBOA + RL approach is developed as an optimized scheduling model which explored and processed diversified numbers of tasks by minimizing latency and energy costs. It employed a multi-objective methodology using the balanced exploration and exploitation capabilities of SBOA with multi-strategy benefits help in maximizing the resource utilization rate and shortening the makespan. The results of the proposed IMSESBOA + RL approach with 5 Cloud VMs and 10 Fog VMs and different IoT tasks minimized the mean makespan by 19.42%, better than the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches. On the other hand, the proposed IMSESBOA + RL approach with respect to 10 Cloud VMs and 20 Fog VMs and different IoT tasks reduced the average makespan by 18.52%, better than the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches. Further the proposed IMSESBOA + RL approach with respect to 15 Cloud VMs and 30 Fog VMs with different IoT tasks reduced the mean makespan by 17.84%, better than the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches. Furthermore, the proposed IMSESBOA + RL approach minimized the energy consumption under 1000 to 5000 tasks with 15 cloud VMs and 30 Fog VMs by 9.68%, compared to the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches used for comparison. The proposed IMSESBOA + RL minimized the scheduling cost under 1000 to 5000 tasks with 15 cloud VMs and 30 Fog VMs by 9.06%, compared to the baseline IMSESBOA, MOMFOA, TSMHGSMPA, MOGWATSS and GAOITSA approaches used for comparison.

Limitations

The proposed IMSESBOA + RL-based IoT TS mechanism face the challenge of extensive data requirements and high demands of computational resources in the fog-cloud environment during the training process. It is also due to the dependency of well-defined reward functions which help in addressing the uncertain and dynamic nature of fog computing scenario.

Future scope of enhancement

The proposed scheduling algorithm can be implemented using the containers instead of VMs such that it can be extended with the suitability of being implemented in the dynamic environments instead of using the VMs of the cloud and fog computing system. It can be further modified in such a way that it can be used for load balancing and workflow schedule determination process. It can also be enhanced using recently developed hybrid optimization techniques, where one component excels in exploration and the other in exploitation. In addition, deep learning approaches can be employed to address the challenge of dynamic scheduling.

Data availability

All data generated or analyzed during this study are included in this article.

Received: 19 January 2025; Accepted: 27 November 2025
Published online: 03 December 2025

References

1. Xu, J., Li, K., Chen, Y. & Huang, J. Optimal task scheduling and resource allocation for Self-Powered sensors in internet of things: an energy efficient approach. *IEEE Trans. Netw. Serv. Manage.* **21** (4), 4410–4420 (2024).
2. Ali, A. et al. Multi-Objective Harris Hawks optimization based task scheduling in Cloud-Fog computing. *IEEE Internet Things J.* **11** (13), 24334–24352 (2024).
3. Kumar, N. J., Praveen, R., Selvaraj, D. & Dhinakaran, D. Hybrid spotted hyena and Whale optimization algorithm-based dynamic load balancing technique for cloud computing environment. *China Commun.* **22** (8), 206–227 (2025).
4. Khezri, E. et al. DLJSF: Data-Locality aware job scheduling IoT tasks in fog-cloud computing environments. *Results Eng.* **21**, 101780 (2024).
5. Daghayeghi, A., & Nickray, M. Delay-aware and energy-efficient task scheduling using strength pareto evolutionary algorithm II in Fog-Cloud Computing paradigm. *Wireless Personal Communications*, **138**(1), 409–457 (2024).
6. Krishna Meera, V., Balasubramanian, C., & Praveen, R. Hybrid Hippopotamus Optimization Algorithm–Based Energy-Efficient Stable Cluster Construction Protocol for Data Routing in VANETs.*International Journal of Communication Systems*, **38**(13), e70171 (2025).
7. Ghafari, R. & Mansouri, N. Fuzzy reinforcement learning algorithm for efficient task scheduling in Fog-Cloud IoT-Based systems. *J. Grid Comput.* **22** (4), 66 (2024).
8. Ebrahim Pourian, R., Fartash, M. & Akbari Torkestani, J. A deep learning model for energy-aware task scheduling algorithm based on learning automata for fog computing. *Comput. J.* **67** (2), 508–518 (2024).

9. Nivitha, K., Pabitha, P. & Praveen, R. CBBM-WARM: A workload-aware meta-heuristic for resource management in cloud computing. *China Commun.* **22** (6), 255–275 (2025).
10. Lin, C. Y. & Liao, W. AoI-Aware interference mitigation for Task-Oriented multicasting in Multi-Cell NOMA networks. *IEEE Trans. Wireless Commun.* **23** (9), 11341–11356 (2024).
11. Alsadie, D. Advancements in heuristic task scheduling for IoT applications in fog-cloud computing: challenges and prospects. *PeerJ Comput. Sci.* **10**, e2128 (2024).
12. Nazemi, S. & Khorsand, R. C-KHCS: Multi-Objective workflow scheduling using chaotic Krill herd optimization and improved cuckoo search in fog computing. *IEEE Trans. Serv. Comput.* **17** (5), 2095–2108 (2024).
13. Reddy, P. B., & Sudhakar, C. IOTD: Intelligent offloading of tasks with deadlines in Edge-Fog-Cloud computing environment using hybrid approach. *Cluster Computing*, **27**(7), 9873–9885 (2024).
14. Karthik, K., Balasubramanian, C. & Praveen, R. Hybrid golden Jackal and moth flame optimization algorithm based coverage path planning in heterogeneous UAV networks. *Sci. Rep.* **15** (1), 31054 (2025).
15. Aburukba, R. O., Landolsi, T. & Omer, D. A heuristic scheduling approach for fog-cloud computing environment with stationary IoT devices. *J. Netw. Comput. Appl.* **180**, 102994 (2021).
16. Kumar, M. S., Reddy, K. G. & Donthi, R. K. SSKHOA: hybrid metaheuristic algorithm for resource aware task scheduling in cloud-fog computing. *Int. J. Inform. Technol. Comput. Sci.* **16** (1), 1–12 (2024).
17. Yakubu, I. Z. & Murali, M. An efficient meta-heuristic resource allocation with load balancing in IoT-Fog-cloud computing environment. *J. Ambient Intell. Humaniz. Comput.* **14** (3), 2981–2992 (2023).
18. Abdel-Basset, M., Moustafa, N., Mohamed, R., Elkomy, O. M. & Abouhawwash, M. Multi-objective task scheduling approach for fog computing. *IEEE Access.* **9**, 126988–127009 (2021).
19. Sing, R. et al. A Whale optimization algorithm-based resource allocation scheme for cloud-fog based Iot applications. *Electronics* **11** (19), 3207 (2022).
20. Ijaz, S., Munir, E. U., Ahmad, S. G., Rafique, M. M. & Rana, O. F. Energy-makespan optimization of workflow scheduling in fog–cloud computing. *Computing* **103**, 2033–2059 (2021).
21. Mousavi, S., Mood, S. E., Souri, A. & Javidi, M. M. Directed search: a new operator in NSGA-II for task scheduling in IoT based on cloud-fog computing. *IEEE Trans. Cloud Comput.* **11** (2), 2144–2157 (2022).
22. Abohamama, A. S., El-Ghamry, A. & Hamouda, E. Real-time task scheduling algorithm for IoT-based applications in the cloud–fog environment. *J. Netw. Syst. Manage.* **30** (4), 54 (2022).
23. Hussain, S. M. & Begh, G. R. Hybrid heuristic algorithm for cost-efficient QoS aware task scheduling in fog–cloud environment. *J. Comput. Sci.* **64**, 101828 (2022).
24. Agarwal, G., Gupta, S., Ahuja, R. & Rai, A. K. Multiprocessor task scheduling using multi-objective hybrid genetic algorithm in Fog–cloud computing. *Knowl. Based Syst.* **272**, 110563 (2023).
25. Saif, F. A., Latip, R., Hanapi, Z. M. & Shafinah, K. Multi-objective grey Wolf optimizer algorithm for task scheduling in cloud-fog computing. *IEEE Access.* **11**, 20635–20646 (2023).
26. Khiat, A., Haddadi, M. & Bahnes, N. Genetic-based algorithm for task scheduling in fog–cloud environment. *J. Netw. Syst. Manage.* **32** (1), 3 (2024).
27. Attiya, I., Abd Elaziz, M. & Issawi, I. An improved hunger game search optimizer based IoT task scheduling in cloud–fog computing. *Internet Things.* **26**, 101196 (2024).
28. Salehnia, T. et al. An optimal task scheduling method in IoT-Fog-Cloud network using multi-objective moth-flame algorithm. *Multimedia Tools Appl.* **83** (12), 34351–34372 (2024).
29. Ali, H. S. & Sridevi, R. Mobility and security aware real-time task scheduling in fog-cloud computing for IoT devices: a fuzzy-logic approach. *Comput. J.* **67** (2), 782–805 (2024).
30. Salimi, R., Azizi, S. & Abawajy, J. A greedy randomized adaptive search procedure for scheduling IoT tasks in virtualized fog–cloud computing. *Trans. Emerg. Telecommunications Technol.*, **35**(5), e4980 (2024).
31. Vijayalakshmi, V. & Saravanan, M. Reinforcement learning-based multi-objective energy-efficient task scheduling in fog-cloud industrial IoT-based systems. *Soft Computing-A Fusion Found. Methodologies Appl.*, **27**(23). 17473 (2023).
32. Wang, Z., Goudarzi, M., Gong, M. & Buyya, R. Deep reinforcement learning-based scheduling for optimizing system load and response time in edge and fog computing environments. *Future Generation Comput. Syst.* **152**, 55–69 (2024).
33. Choppara, P. & Mangalampalli, S. S. A hybrid task scheduling technique in fog computing using fuzzy logic and deep reinforcement learning. *IEEE Access.* **12**, 176363–176388 (2024).
34. Fu, Y., Liu, D., Chen, J. & He, L. Secretary bird optimization algorithm: a new metaheuristic for solving global optimization problems. *Artif. Intell. Rev.* **57** (5), 1–102 (2024).
35. Chilamkurthy, N. S. et al. Energy-efficient and qos-aware data transfer in q-learning-based small-world Lpwans. *IEEE Internet Things J.* **10** (24), 22636–22649 (2023).
36. Sagar, A. S., Haider, A. & Kim, H. S. A hierarchical adaptive federated reinforcement learning for efficient resource allocation and task scheduling in hierarchical IoT network. *Comput. Commun.* **229**, 107969 (2025).
37. Askar, N. A. & Habbal, A. *RLEAFS: Reinforcement Learning Based Energy Aware Forwarding Strategy for NDN Based IoT Networks* (IEEE Access, 2024).
38. Choppara, P. & Lokesh, B. *Efficient Task Scheduling and Load Balancing in Fog Computing for Crucial Healthcare Through Deep Reinforcement Learning* (IEEE Access, 2025).
39. C., Balasubramanian G., Sathya R., Praveen (2024) Hybrid Mexican axolotl and bitterling fish optimization algorithm-based spectrum sensing multi-hop clustering routing protocol for cognitive sensor networks Scientific Reports 14(1) 10.1038/s41598-024-82311-z
40. M., Ananthi K., Valarmathi A., Ramathilagam R., Praveen (2025) Hybrid lion and exponential PSO-based metaheuristic clustering approach for efficient dynamic data stream management Scientific Reports 15(1) 10.1038/s41598-025-07404-9

Author contributions
All the authors contribute equally to the paper.

Funding
Not Applicable.

Declarations

Competing interests
The authors declare no competing interests.

Ethical approval
This article does not contain any studies with human participants or animals performed by any of the authors.

Consent to publish
Not Applicable.

Additional information
Correspondence and requests for materials should be addressed to K.S.